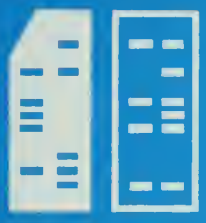
Report No. 359

AN ALGORITHM FOR THE MINIMIZATION OF TWO-LEVEL MULTIPLE-OUTPUT NETWORKS

by

Frank Tuan-Lin Chen

February, 1970



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

Report No. 359

AN ALGORITHM FOR THE MINIMIZATION OF TWO-LEVEL MULTIPLE-OUTPUT NETWORKS *

by

Frank Tuan-Lin Chen

February 1970

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

* This work was submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science in the Graduate College of the University of Illinois, February, 1970.

ACKNOWLEDGMENT

The author expresses his grateful thanks to his thesis advisor, Professor Saburo Muroga, for his invaluable and continuous guidance, encouragement, and constructive suggestions during the preparation of this thesis. He also wishes to thank Mr. T. Nakagawa for his comments and suggestions, Mr. A. C. Tillman for his corrections in the first draft, his family and Miss Lily Chang for their encouragement throughout the entire course of the investigation of this thesis. Thanks are also extended to Mrs. K. Hicks who typed the most part of this thesis.

The support of the Department of Computer Science, University of Illinois is also gratefully acknowledged.

TABLE OF CONTENTS

AN ALGORITHM FOR THE MINIMIZATION OF TWO-LEVEL MULTIPLE-OUTPUT NETWORKS

Frank Tuan-Lin Chen, Ph.D.
Department of Computer Science
University of Illinois, 1970

Quine-McCluskey's minimization technique of 2-level multiple-output networks has been widely accepted in the field of logical design. Their approach is based on a special restriction of a 2-level AND-OR network, i.e., all AND gates in the first level and all OR gates in the second level. If such restriction is removed, their approach will no longer be able to reach the minimality.

This dissertation presents an algorithm which is an improvement of the classical Quine-McCluskey minimization technique when the above restriction on a two-level AND-OR network is removed. A minimal network which is less costly than that obtained by Quine-McCluskey's method can be obtained by this new algorithm. The new approach is based on the refinement of the classical multiple-output prime implicant (mopi). This refinement enables us to consider all the possibilities of cost saving in a two-level AND-OR network.

Some of the algebraic properties of such refinement are explored. Properties of the minimal network obtained by this new approach are also investigated. A theorem which corresponds to the McCluskey multiple-output prime implicant theorem is presented. The integer linear programming method is employed to solve the minimal covering problem. From the solution of the minimal covering problem the minimal network is to be constructed.

The inclusion of inversions which can be obtained with and without extra cost is investigated, and the minimization procedure when inversions are included is also presented. Finally, the minimization with don't-cares is presented which is a modification of the minimization without don't-cares.

1. INTRODUCTION

1.1 Introduction

It is not uncommon that a logical circuit designer encounters problems of logical network design for several outputs rather than a single output. Under a certain cost criterion, the design of multiple-output networks of the least cost is much more difficult than the design of single-output networks of the least cost. The reason for this is that the complexity of gate interconnections within a multiple-output network makes the minimal realization difficult to reach. It is unlike the design of a single-output network whose least cost network can be achieved if the simplest Boolean expression of such a function can be found and if the network is to be realized within two levels with AND and OR gates. In the design of a multiple-output network, even if the simplest Boolean expression of each individual function is used, the network which combines the single-output networks corresponding to these expressions is still not necessarily of the least cost because of the possibility that more than one output can share a gate in a multiple-output network.

There are some well-known methods for finding a minimal multiple-output network, where the minimality means the least cost under a specified cost criterion. Among them, the McCluskey minimization method[22] which employs the concept of prime implicant to design a minimal two-level multiple-output network is most widely known. His two-level networks are restricted to a special type of gate configuration in two levels, in which the first level gates are all AND gates feeding to all OR output gates. The minimality may be improved if such restriction on gate configuration is removed; that is, the network may have a mixture of AND and OR gates in its first level as well as in its output level. In this paper an approach is described for solving the problem of designing a minimal two-level, multiple-output network without such restrictions on the gate configuration. This approach may be

considered as an extension of McCluskey's method because his prime implicant concept is used and extended.

The term "network" used throughout the text of this paper will mean a loop-free combinational multiple-output network of two levels. The logical elements used in the construction of a network are restricted to AND and OR gates. The network designed with the inclusion of NOT gates will also be discussed. It is assumed at this point that no restriction is placed on the number of fan-in's and the number of fan-out's for any gate in a network.

There are three cases which will be treated in this paper:

1. Minimization of a network for a given set of output functions which are all completely specified with no inversion inside the network. The inversions are allowed only at the inputs. Chapters 2 and 3 will be devoted to the discussion of the minimization method for this case.

2. Minimization of a network for a given set of output functions which are all completely specified, permitting the use of inversions inside the network. The inversions are obtained either through the use of NOT gates or through a special type of AND/OR gates which provide complemented as well as uncomplemented outputs. This is treated in Chapter 4.

3. Minimization of a network for a given set of output functions which are incompletely specified with no inversion permitted inside the network. This is treated in Chapter 5.

1.2 Definitions and Notations

A Boolean function f of n-variable $x_1, x_2, \ldots, x_n$ is denoted by $f(\vec{x})$ where $\vec{x} = (x_1, x_2, \ldots, x_n)$. $f(\vec{x})$ and x_i, $i = 1, 2, \ldots, n$ are either 1 or 0. If more than one Boolean function is encountered, the notations $f_1, f_2, \ldots, f_m$ will be used. The letters f and f_i will be exclusively used as outputs of a network. Thus, f and f_i are also called output functions.

The complement of a variable x_i is written as $\bar{x}_i$. The complement of a function f_i is written as $\bar{f}_i$. The letters x_i and $\bar{x}_i$ are called literals for any i = 1, 2, ..., n. x_i and $\bar{x}_i$ are two different literals. x_i and $\bar{x}_i$ are used as inputs of a network. Thus, they are also called input variables. The star sign (*) is used to denote a complemented or uncomplemented variable or function. For example, $f_i^* = f_i$ or $\bar{f}_i$ and $x_i^* = x_i$ or $\bar{x}_i$.

The disjunction (symbol $\vee$) and the conjunction (symbol $\wedge$, or $\cdot$) of variables are as defined conventionally. They are realized by an "OR" gate and an "AND" gate, respectively. Boolean functions are also defined by Boolean forms, which are denoted by capitol Greek letters Φ_i. A Boolean function f_i defined by a Boolean form Φ_i is denoted as $f_i = \Phi_i$. There may be many Boolean forms which define the same Boolean function. For example, two Boolean forms $\Phi = x_1 x_3 \vee x_3 x_4$ and $\Phi' = x_3 (x_1 \vee x_4)$ define the same function f.

Some basic definitions which will be used later are given in the following:

Definition 1.2.1: A term is a product of literals where a literal for each variable appears at most once. An alterm is a disjunction of literals where a literal for each variable appears at most once.

Definition 1.2.2: A term denoted by T_1 is said to subsume another term denoted by T_2 if T_1 has all the literals of T_2. An alterm denoted by A_1 is said to subsume another alterm denoted by A_2 if and only if A_1 has all the literals of A_2.(13)

For example, a term $x_1 \bar{x}_2 x_3$ subsumes another term $x_1 x_3$, and an alterm $x_1 \vee \bar{x}_2 \vee x_3$ subsumes another alterm $x_1 \vee x_3$. T or T_i will be used exclusively to denote terms, and A or A_i to denote alterms. In some special cases,

T or T_i may be a single literal term. A or A_i may be a single literal alterm.

It should be noted that the complement of a term T (i.e., $\bar{T}$) is an alterm, and that the complement of an alterm A (i.e., $\bar{A}$) is a term. For example, if $T = x_1 x_3$ is a term, then $\bar{T} = \bar{x}_1 \vee \bar{x}_3$ is an alterm; and if $A = x_1 \vee \bar{x}_2 \vee x_3$ is an alterm, then $\bar{A} = \bar{x}_1 x_2 \bar{x}_3$ is a term.

Definition 1.2.3: An OR-JOINT is a Boolean form which is a disjunction of terms, and/or alterms. An AND-JOINT is a Boolean form which is a conjunction of alterms, and/or terms.

If Φ is an OR-JOINT, then Φ can be expressed as

$$\Phi = T_1 \vee \ldots \vee T_i \vee A_1 \vee \ldots \vee A_j,$$

where $T_1, \ldots, T_i$ are terms and $A_1, \ldots, A_j$ are alterms. If Φ is an AND-JOINT, then Φ can be expressed as

$$\Phi = A_1 \wedge \ldots \wedge A_i \wedge T_1 \wedge \ldots \wedge T_j,$$

where $A_1, \ldots, A_i$ are alterms and $T_1, \ldots, T_j$ are terms.

For example, the Boolean form $x_1 x_2 \vee (x_3 \vee x_4)$ is an OR-JOINT, which is a disjunction of a term $x_1 x_2$ and an alterm $(x_3 \vee x_4)$, and the Boolean form $x_2 x_3 (\bar{x}_1 \vee x_4)$ is an AND-JOINT which is a conjunction of a term $x_2 x_3$ and an alterm $(\bar{x}_1 \vee x_4)$.

In the following, the structures of a term and an alterm will be defined, and then the structures of an OR-JOINT and an AND-JOINT are defined.

Definition 1.2.4: The structure of a term $T = x^*_{i_1} x^*_{i_2} \ldots x^*_{i_k}$ means an "AND" gate with inputs $x^*_{i_1}, x^*_{i_2}, \ldots, x^*_{i_k}$ and output T. The structure of an alterm $A = x^*_{i_1} \vee x^*_{i_2} \vee \ldots \vee x^*_{i_k}$ means an "OR" gate with inputs $x^*_{i_1}, x^*_{i_2}, \ldots, x^*_{i_k}$ and output A.

Definition 1.2.5: The structure of an OR-JOINT $\Phi = T_1 \vee \ldots \vee T_i \vee A_1 \vee \ldots \vee A_j$ means an "OR" gate with inputs $T_1, \ldots, T_i$ $A_1, \ldots, A_j$ and output Φ, where $T_1, \ldots, T_i$ are outputs of the structures of terms $T_1, \ldots, T_i$ and $A_1, \ldots, A_j$ are outputs of the structures of alterms $A_1, \ldots, A_j$, respectively.

Definition 1.2.6: The structure of an AND-JOINT $\Phi = A_1 \wedge \ldots \wedge A_i \wedge T_1 \wedge \ldots \wedge T_j$ means an "AND" gate with inputs $A_1, \ldots, A_i$ $T_1, \ldots, T_j$ and output Φ, where $A_1, \ldots, A_i$ are the outputs of the structures of alterms $A_1, \ldots, A_i$ and $T_1, \ldots, T_j$ are the outputs of the structures of terms $T_1, \ldots, T_j$, respectively.

Based on the above definitions of the structures of a term, an alterm, an OR-JOINT and an AND-JOINT one can define a two-level, multiple-output network.

Definition 1.2.7: A two-level, multiple-output network realizing a given set of output functions $f_1, f_2, \ldots, f_m$ is denoted by $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ where $f_i = \Phi_i$ for $i = 1, 2, \ldots, m$, and each Φ_i is either an OR-JOINT or an AND-JOINT. The network consists of AND and OR gates in two levels with $x_1, \bar{x}_1, x_2, \bar{x}_2, \ldots, x_n, \bar{x}_n$ as inputs and $f_1, f_2, \ldots, f_m$ as outputs.

The redundant gate is defined in the following.

Definition 1.2.8: An AND (OR) gate in a network is said to be redundant if there exists another AND (OR) gate in the same network whose output is identical to that of the former one. A network is said to be irredundant if there exists no redundant gate within that network.

Definition 1.2.9: A function f is said to imply another function g if whenever $f = 1$ implies $g = 1$. It is denoted by $f \subseteq g$. f and g are said to be comparable if either $f \subseteq g$ or $f \supseteq g$ holds. f and g are said to be incomparable otherwise.

As mentioned before this study is confined to two-level, loop-free combinational networks. The "two-level" here means that the maximum number of gates over all strings of gates from the inputs to the outputs of a network is two.

1.3 Cost Function

The objective of any simplification or minimization procedure is to obtain a circuit which performs the desired logical operation at the smallest cost under some minimality criteria. Different criteria of minimality have been defined by different authors, e.g.,

1. The total number of gate-inputs, i.e., the sum of the number of inputs of all gates in a network.
2. The total number of logical gates in a network.

Many other criteria have been used as minimality criteria, such as, the minimization of the total number of packages if the circuits are constructed with standard gate packages which are prewired for a fixed number of inputs, and the minimization of the total number of diodes if the circuits are to be built by interconnecting diodes and resistors.

At the advent of integrated circuitry, the following criteria may be important.

1. The total length of interconnections among gates. This effectively influences the physical size and the complexity of an integrated circuit chip.
2. The number of cross-over interconnections on an integrated circuit chip.
3. The total number of input terminal pins on an integrated circuit chip.

However, it is very hard to incorporate the above criteria all at one time into any minimization procedure, especially in the design of multiple-output networks. For simplicity sake, only two of them will be incorporated into the minimization procedure of this study, i.e., the total number of gate-inputs and the total number of logical gates in a network. If the number of gate-inputs is N_1 and the number of gates is N_2, then the cost function of the minimization procedure can be defined as $W_1 N_1 + W_2 N_2$ where W_1 and W_2

are weight factors. For all practical purposes $W_1 = W_2 = 1$ will be set. The cost function is then simply $N_1 + N_2$. This may roughly represent the construction cost of a network and meanwhile simplify the minimization procedure which will be described later.

A minimal network of multiple-output under the cost function defined above is defined in the following.

Definition 1.3.1: A network realizing a given set of output functions is said to be a minimal multiple-output network if it realizes the same set of output functions with the minimum sum of the total number of gate-inputs and the total number of gates in the network.

It is obvious that in a minimal network of AND and OR gates there is no gate with a single input and that every minimal network must be an irredundant network.

2. PRELIMINARY CONCEPTS

2.1 Introduction

The procedure for minimization of a two-level logical network has been investigated by many authors. A new procedure will be presented in this paper which is primarily motivated by the paper entitled "Discussion of Some Flaws in the Classical Theory of Two-Level Minimization of Multiple-Output Switching Networks"[39] by Weiner and Dwyer. The restriction of gate configuration (i.e., all AND gates in the first level and all OR gates in the second output level) existing in the classical two-level minimization technique is removed in the new procedure. This is done by redefining and refining the "prime implicant" which is the key concept upon which the classical two-level minimization technique is based.

2.2 Some New Definitions

A two-level network minimization procedure derived first by Quine[33] and improved by McCluskey[21] is widely known in switching theory. Their approach is based on a set of prime implicants which are generated from a set of given functions. Prime implicants are generated in such a way that a minimal network can be designed. However, if the restriction (i.e., all AND gates in the first level and all OR gates in the second level) is removed, their approach does not lead to an optimal network as has been pointed out by Weiner and Dwyer. In order to illustrate this an example is given in the following.

A switching network of three outputs f_1, f_2, f_3 is obtained by the McCluskey method and is shown in Figure 2.1.

A realization which has a fewer number of gates and gate-inputs is shown in Figure 2.2.

In Figure 2.1, the network consists of 8 gates and 28 gate-inputs, and the cost is 36. In Figure 2.2 however, the network consists of only 7 gates and 18 gate-inputs, and the cost is 25.

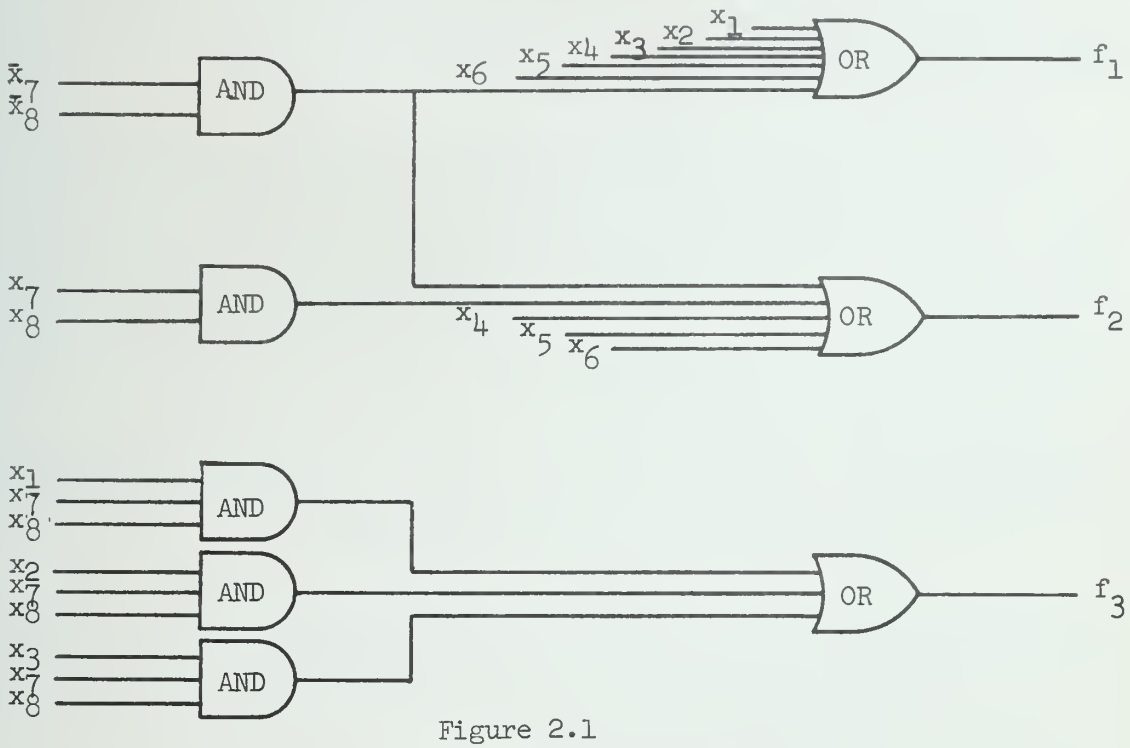


Figure 2.1

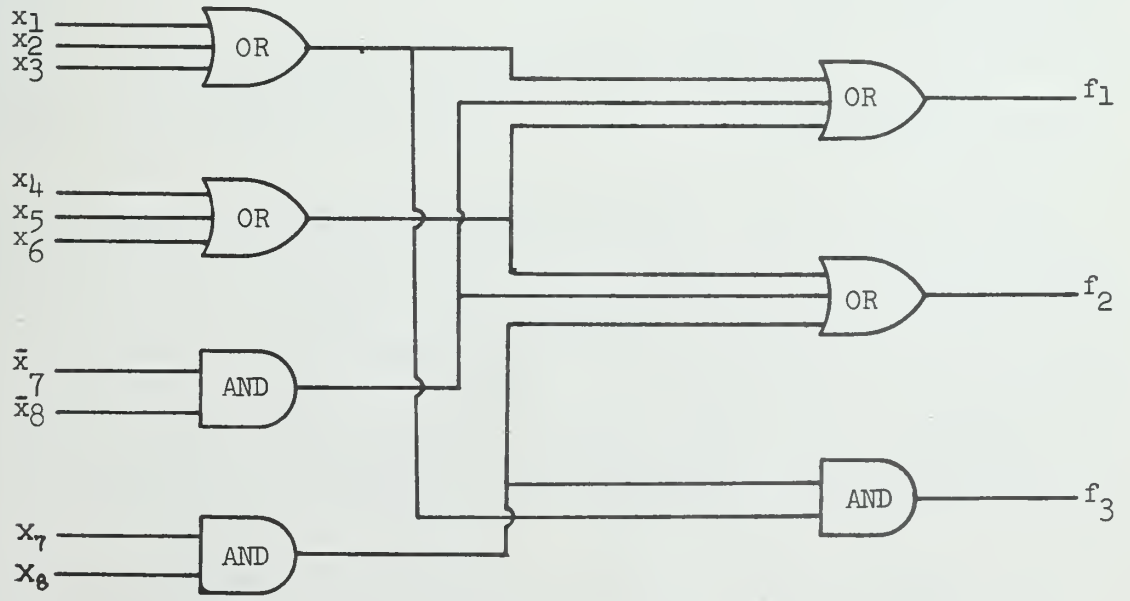


Figure 2.2

Figure 2.2 shows that the OR gate producing the alterm $(x_1 \vee x_2 \vee x_3)$ is shared by f_1 and f_3, the OR gate producing the alterm $(x_4 \vee x_5 \vee x_6)$ is shared by f_1 and f_2, and the AND gates producing $x_7\ x_8$ and $\bar{x}_7\ \bar{x}_8$, respectively, are shared by two functions. Those gates shared by more than one output function do not exist in Figure 2.1.

Therefore, in order to take into account the possibility that gates can be shared by more than one output function, new definitions for the prime implicant as stated in the paper by Weiner and Dwyer should be introduced. The following definitions are given to facilitate the design procedure with the consideration discussed above.

Definition 2.2.1: A term $T = x^*_{i_1}\ x^*_{i_2} \ \ldots\ x^*_{i_k}$ is called a prime implicant of f if the following two conditions are satisfied:

(1) T implies f, i.e., whenever T = 1, f = 1.

(2) If any literal in T is removed from T, then the remaining term no longer implies f.

Definition 2.2.2: A term $T = x^*_{i_1}\ x^*_{i_2} \ \ldots\ x^*_{i_k}$ is called a multiple-output prime implicative term of the set of functions $f_1, f_2, \ldots, f_m$ if T is a prime implicant of the product of functions, either $f_{j_1} \wedge f_{j_2} \wedge \ldots \wedge f_{j_\ell}$ or $\bar{f}_{j_1} \wedge \bar{f}_{j_2} \wedge \ldots \wedge \bar{f}_{j_\ell}$ for some set $\{j_1, j_2, \ldots, j_\ell\} \in \{1, 2, \ldots, m\}$, and $\ell \geq 1$.

Note that the only difference between the conventional multiple-output prime implicant and the multiple-output prime implicative term defined above is the inclusion of the product of complemented functions in the definition.

Definition 2.2.3: An alterm $A = x^*_{i_1} \vee x^*_{i_2} \vee \ldots \vee x^*_{i_k}$, $k \geq 2$, is called a multiple-output prime implicative alterm of the set of functions $f_1, f_2, \ldots f_m$ if

(1) alterm A implies some product of functions, either $f_{j_1} \wedge f_{j_2} \wedge \ldots f_{j_\ell}$ or $\bar{f}_{j_1} \wedge \bar{f}_{j_2} \wedge \ldots \wedge \bar{f}_{j_\ell}$, where $\{j_1, j_2, \ldots, j_\ell\}$ is a subset

{1, 2, ..., m}, and

(2) there exists no other alterm of more literals which subsumes A and also satisfies the above condition (1).

Note in the above definition if k = 1, the definition becomes the case of Definition 2.2.2.

In Figure 2.2, for example, $(x_4 \vee x_5 \vee x_6)$ is a multiple-output prime implicative alterm of f_1, f_2, f_3 for the following reason. $(x_4 \vee x_5 \vee x_6)$ satisfies condition (1) with respect to $f_1 \wedge f_2$, because $(x_4 \vee x_5 \vee x_6)$ implies $f_1 \cdot f_2 = (x_4 \vee x_5 \vee x_6) \vee (x_1 \vee x_2 \vee x_3)(x_7 x_8) \vee \bar{x}_7 \bar{x}_8$. $(x_4 \vee x_5 \vee x_6)$ also satisfies condition (2) because there is no other alterm of more literals which subsumes $(x_4 \vee x_5 \vee x_6)$ and also implies the product of functions $f_1 \cdot f_2$.

Definition 2.2.4: A term $T = x_{i_1}^* x_{i_2}^* \ldots x_{i_k}^*$ is called a multiple-output implicative co-term (the abbreviation of the correlated term) for a set of output functions $f_1, f_2, \ldots, f_m$ if there exists two disjoint subsets of functions $(f_{j_1}, f_{j_2}, \ldots, f_{j_\ell})$ and $(f_{p_1}, f_{p_2}, \ldots, f_{p_q})$ of $(f_1, f_2, \ldots, f_m)$ such that

(1) the term T implies the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$, and

(2) the alterm $\bar{T}$ implies the product of the complemented functions $\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$ for some $\ell \geq 1$, $q \geq 1$.

In the above definition the conditions $\ell \geq 1$ and $q \geq 1$ are necessary. The conditions $\ell \geq 1$ and $q \geq 1$ are equivalent to the conditions $q < m$ and $\ell < m$. The condition $\ell < m$ must hold because if T implies $f_1 f_2 \ldots f_m$ of condition (1), then the product in condition (2) becomes null. The condition $q < m$ must hold because if $\bar{T}$ implies $\bar{f}_1 \bar{f}_2 \ldots \bar{f}_m$ of condition (2), then the product in condition (1) becomes null.

Also, note that if q = 1, the above condition (2) of Definition 2.2.4 can be simply stated as "the term T is implied by f_{p_1}" since "$\bar{T}$ implies $\bar{f}_{p_1}$" is equivalent to "T is implied by f_{p_1}."

As an example, the term $x_7\ x_8$ in Figure 2.2 is a multiple-output implicative co-term since $x_7\ x_8$ implies the function f_2, and $\bar{x}_7 \vee \bar{x}_8$ implies the complemented function $\bar{f}_3$. Here, the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ in Definition 2.2.4 is simply f_2, the product of the complemented functions $\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$ is simply $\bar{f}_3$, and $\ell = 1$, q = 1.

Definition 2.2.5: An alterm $A = x^*_{i_1} \vee x^*_{i_2} \vee \ldots \vee x^*_{i_k}$ for $k \geq 2$ is called a multiple-output implicative co-alterm (the abbreviation of the correlated alterm) for a set of output functions $f_1, f_2, \ldots, f_m$ if there exists two disjoint subsets of functions $(f_{j_1}, f_{j_2}, \ldots, f_{j_\ell})$ and $(f_{p_1}, f_{p_2}, \ldots, f_{p_q})$ of $(f_1, f_2, \ldots, f_m)$ such that

(1) the alterm A implies the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$, and

(2) the term $\bar{A}$ implies the product of the complemented functions $\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$, for some ℓ, $q \geq 1$.

As in Definition 2.2.4, the conditions $\ell \geq 1$ and $q \geq 1$, and accordingly $\ell < m$, $q < m$ are also necessary. And, if q = 1, condition (2) of the above definition can be simply stated as "the alterm A is implied by a function f_{p_1}."

For example, the alterm $(x_1 \vee x_2 \vee x_3)$ in Figure 2.2 is a multiple-output implicative co-alterm since $(x_1 \vee x_2 \vee x_3)$ implies f_1 and $(\overline{x_1 \vee x_2 \vee x_3})$ implies $\bar{f}_3$. Here, the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ is simply f_1 and $\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$ is simply $\bar{f}_3$ in this example.

The conditions given in Definitions 2.2.2 to 2.2.5 for the multiple-output prime implicative term and others will be explained in the next section. The motivation for introducing the concept of the multiple-output implicative co-term and co-alterm will also be discussed. As will be seen later, the multiple-output prime implicative alterms, multiple-output implicative co-terms

and co-alterms can all be generated from the set of multiple-output prime implicative terms. Thus, we have the following definition.

Definition 2.2.6: The collection of all multiple-output prime implicative terms of the set of functions $f_1, f_2, \ldots, f_m$ is called the basic set of multiple-output implicative elements.

For simplicity sake, multiple-output prime implicative terms and alterms will be abbreviated as MOPI terms and MOPI alterms, respectively. The multiple-output implicative co-term and multiple-output implicative co-alterm will be abbreviated as MOCO term and MOCO alterm, respectively. The term "MO elements" will be used as a collective name for all of the above. Accordingly, the basic set of multiple-output implicative elements is abbreviated as the basic set of MO elements.

The term "a MOPI term (or MOPI alterm) with respect to $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ (or $\bar{f}_{j_1}, \bar{f}_{j_2}, \ldots, \bar{f}_{j_\ell}$)" will be used to mean that $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ (or $\bar{f}_{j_1}, \bar{f}_{j_2}, \ldots, \bar{f}_{j_\ell}$) is the subset of functions used in the definition of a MOPI term (or MOPI alterm). Similarly, the term "a MOCO term (or MOCO alterm) with respect to $(f_{j_1}, f_{j_2}, \ldots, f_{j_\ell})$ and $(f_{p_1}, f_{p_2}, \ldots, f_{p_q})$" will be used to mean that $(f_{j_1}, f_{j_2}, \ldots, f_{j_\ell})$ and $(f_{p_1}, f_{p_2}, \ldots, f_{p_q})$ are the two subsets of functions used in the definition of a MOCO term (or MOCO alterm).

2.3 Heuristic Remarks

This section explains heuristically motivations of the concepts of MOPI term, MOCO term, MOPI alterm, and MOCO alterm.

Consider the possibility that an AND/OR gate which produces a term/an alterm may be shared by more than one output function in a two-level AND-OR minimal cost network. The "minimal cost" here is important because this is what this study is interested in. There are three cases:

Case 1. An AND/OR gate is shared by a set of output functions which are all expressed in OR-JOINT in a minimal network.

Case 2. An AND/OR gate is shared by a set of output functions which are all expressed in AND-JOINT in a minimal network.

Case 3. An AND/OR gate is shared by a set of output functions which are all expressed in AND-JOINT and also by another set of output functions which are all expressed in OR-JOINT in a minimal network.

Each of these three cases will be discussed and illustrated. The case for an AND gate and for an OR gate will be discussed separately.

An AND gate in Case 1 is illustrated in the following Figure 2.3 which shows a part of a minimal network.

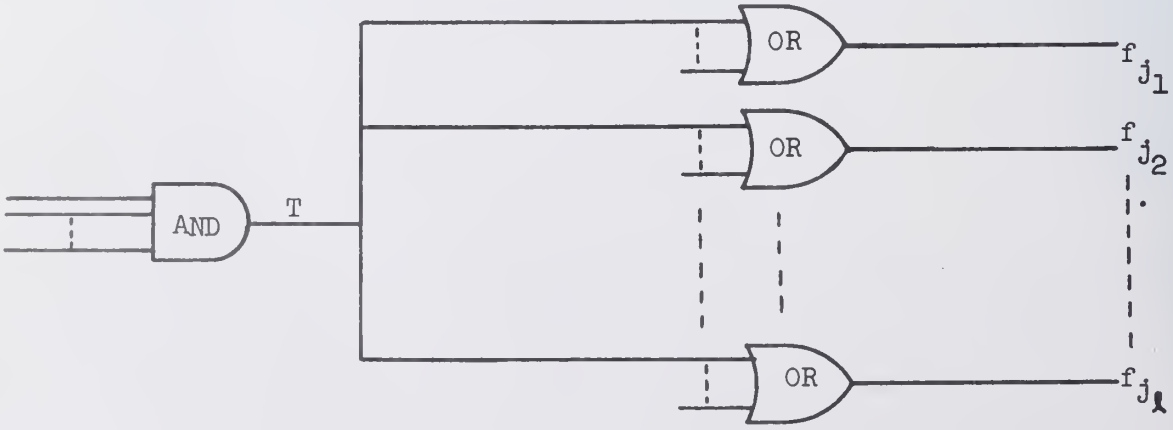


Figure 2.3

Here f_{j_1}, f_{j_2}, ..., f_{j_ℓ} are the set of output functions in the minimal network which share the AND gate. Each of f_{j_1}, f_{j_2}, ..., f_{j_ℓ} is expressed in OR-JOINT. In this case, the term T produced by the AND gate must be a prime implicant of the product $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$. This is because that whenever T = 1, the product $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} = 1$, and that if T is not a prime implicant of the product $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$, then some input of the AND gate can be removed and a network of lower cost results which is a contradiction since Figure 2.3 is a part of a minimal network. Thus, the term T produced by the AND gate is a MOPI term which satisfies the condition of Definition 2.2.2.

For an OR gate in Case 1, it is illustrated in Figure 2.4 which shows a part of a minimal network.

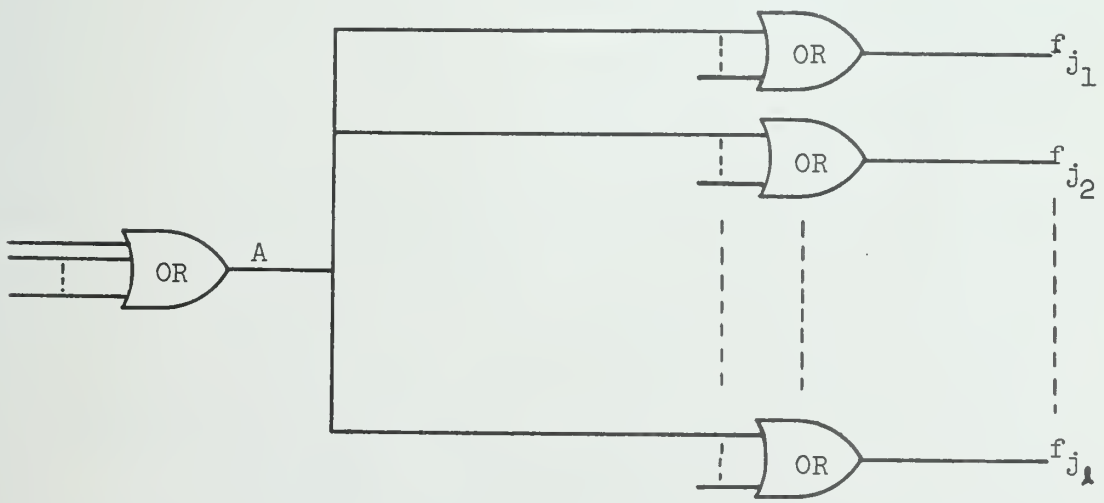


Figure 2.4

The OR gate produces an alterm A for the set of functions f_{j_1}, f_{j_2}, ..., f_{j_ℓ} which share this OR gate. The alterm A must consist of all single literals which imply every one of f_{j_1}, f_{j_2}, ..., f_{j_ℓ} and accordingly the product $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$. This is because that (1) whenever A = 1, $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} = 1$, and (2) if A does not consist of all single literals of the

product $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$, then there exists at least one literal which is an input to all of the output gates (OR gates) of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$. This common input to all OR output gates of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ can all be deleted and then added to the input of the OR gate which produces the alterm A. This results in a network of lower cost which is a contradiction since Figure 2.4 is a part of a minimal network. Thus the alterm A is a MOPI alterm which satisfies the conditions of Definition 2.2.3.

Figure 2.5 which shows a part of a minimal network is an illustration of an AND gate in Case 2.

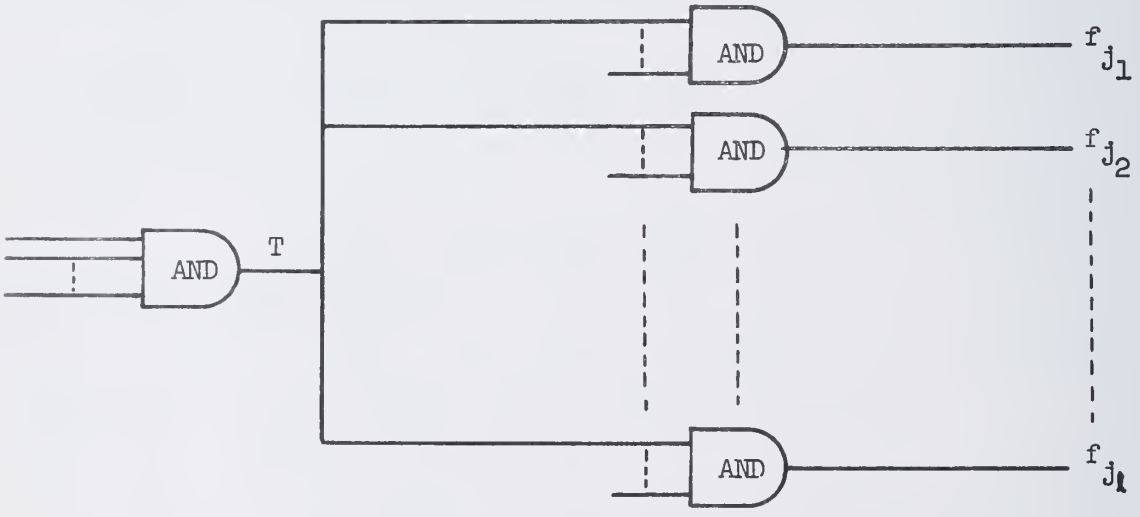


Figure 2.5

The AND gate which produces the term T is shared by a set of output functions $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ which are expressed in AND-JOINT.

It is to be shown that the term T must consist of <u>all</u> literals which are simultaneously implied by all of the functions $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$. This is because of the following two reasons: (1) T must consist of literals which are simultaneously implied by all of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$, since $T = 1$ whenever any one of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ is 1. (2) If T does not consist of all literals which are simultaneously implied by all of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$, then there exists at least one common literal which is an input to all of the

AND output gates of f_{j_1}, f_{j_2}, ..., f_{j_ℓ} (see Figure 2.5). This common input can be deleted from all AND output gates of f_{j_1}, f_{j_2}, ..., f_{j_ℓ} and is added to the input of the AND gate which produces the term T. This results in a network of lower cost which is a contradiction since Figure 2.5 is a part of a minimal network. This shows that T must consist of all literals which are simultaneously implied by all of f_{j_1}, f_{j_2}, ..., f_{j_ℓ}.

From the reason (1) above, it follows that $f_{j_1} \vee f_{j_2} \vee \ldots \vee f_{j_\ell} = 0$ whenever T = 0. In other words, $\bar{T}$ which is a disjunction of single literals implies $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$. However, T consists of <u>all</u> literals each of which is simultaneously implied by all of f_{j_1}, f_{j_2}, ..., f_{j_ℓ}. It follows that $\bar{T}$ which is an alterm is a disjunction of exactly <u>all</u> single literals which imply the product of the complement of functions $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$. This idea can be clarified by the following example. Let $T = x_1 x_2$ be a term consisting of all literals (i.e., x_1 and x_2) which are simultaneously implied by two functions $f_1 = x_1 x_2 x_3$ and $f_2 = x_1 x_2 x_5$. Then the alterm $\bar{T} = \bar{x}_1 \vee \bar{x}_2$ is a disjunction of exactly all single literals (i.e., $\bar{x}_1$ and $\bar{x}_2$) which imply the product of complemented functions $\bar{f}_1 \cdot \bar{f}_2 = \bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \bar{x}_5$. It concludes from the above argument that $\bar{T}$ which is an alterm must be a MOPI alterm which satisfies both the "OR" part of condition (1) and condition (2) of Definition 2.2.3 since the alterm $\bar{T}$ implies $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$ (condition (1)) and also there exists no other alterm of more literals which subsumes $\bar{T}$ and also implies $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$ (condition (2)).

An OR gate in Case 2 is illustrated by Figure 2.6.

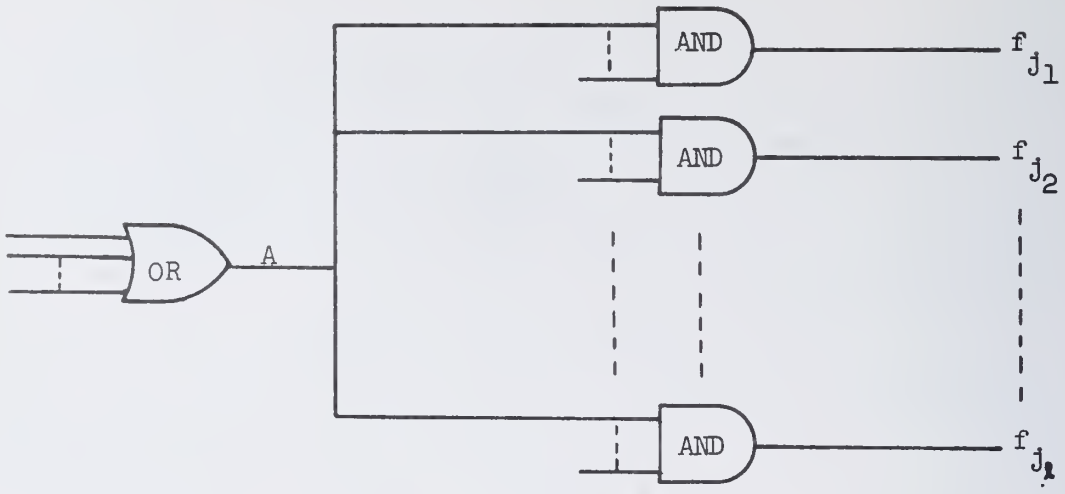


Figure 2.6

$f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ is the set of functions which share the OR gate that produces the alterm A.

The alterm A must be simultaneously implied by $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ because whenever any one of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ is 1, A = 1. This means that the term which is the complement of A (i.e., $\bar{A}$) must imply the product of the complement of functions $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$, because $f_{j_1} \vee f_{j_2} \vee \ldots \vee f_{j_\ell} = 0$ whenever A = 0. It will be proved in the following paragraph that $\bar{A}$ must be a prime implicant of $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$. If so, $\bar{A}$ must be a MOPI term which satisfies the "OR" part of the condition of Definition 2.2.2.

If $\bar{A}$ is not a prime implicant of $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$, then some literal(s) in $\bar{A}$ can be deleted such that the resulting term is a prime implicant of the product $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$. In other words, some input(s) of the OR gate which produces A can be removed which results in a network of lower cost. But, this is a contradiction since Figure 2.6 is a part of a minimal network.

Next, the use of the MOCO term and the MOCO alterm which occurs in Case 3 will be discussed.

For an AND gate in Case 3, Figure 2.7 depicts the interconnections among the AND gate and the output gates (OR gates and AND gates).

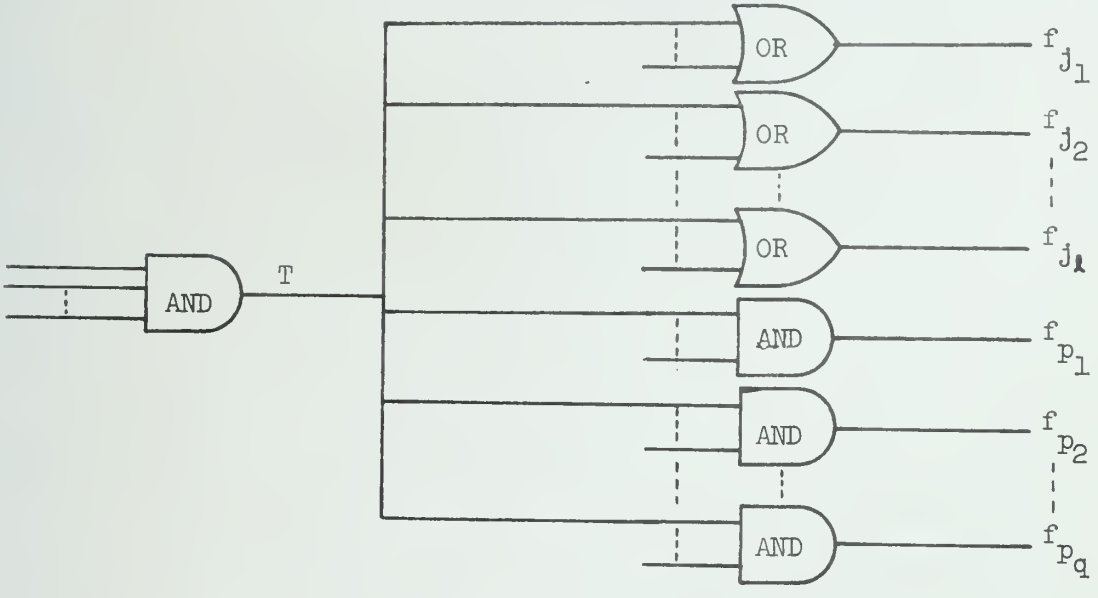


Figure 2.7

$f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ are the set of functions expressed in OR-JOINT and $f_{p_1}, f_{p_2}, \ldots, f_{p_q}$ are the set of functions expressed in AND-JOINT which share the AND gate that produces the term T.

From Figure 2.7, it is clear that the term T must imply functions $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$, and accordingly the product $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$. Also, the term T must be simultaneously implied by functions $f_{p_1}, f_{p_2}, \ldots, f_{p_q}$ (i.e., $f_{p_1} \vee f_{p_2} \vee \ldots \vee f_{p_q} = 0$ whenever $T = 0$) and accordingly $\overline{T}$ implies the product $\overline{f}_{p_1} \cdot \overline{f}_{p_2} \cdot \ldots \cdot \overline{f}_{p_q}$. These are exactly what are required for a MOCO term in Definition 2.2.4. Thus, T in Figure 2.7 is a MOCO term.

A MOCO term does not necessarily be a MOPI term with respect to the set of functions $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$. Also, the complement of a MOCO term does not necessarily be a MOPI alterm with respect to the set of functions $\overline{f}_{p_1}, \overline{f}_{p_2}, \ldots, \overline{f}_{p_q}$. An example of a MOCO term which is not a MOPI term and an example of the complement of a MOCO term which is not a MOPI alterm are shown

in Appendix A.

If the AND gate which produces T in Figure 2.7 is replaced by an OR gate, one has the last case that an OR gate is shared by two sets of functions, one set $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ is in OR-JOINT, and the other set $f_{p_1}, f_{p_2}, \ldots, f_{p_q}$ is in AND-JOINT. A similar explanation to that for the case of Figure 2.7 can be made to show that the alterm produced by the OR gate which is shared by two sets of functions is a MOCO alterm which satisfies the conditions in Definition 2.2.5.

Since an AND/OR gate can be shared either by a set of functions which is expressed in OR-JOINT, or by a set of functions which is expressed in AND-JOINT, or by both, the above discussion explores and exhausts all possibilities in which a gate can be shared by more than one output function in the minimal cost two-level network of AND and OR gates. Hence, the consideration of MOPI terms, MOPI alterms, MOCO terms, and MOCO alterms is equivalent to the consideration of all possible gates which can be shared by more than one output function.

Thus, when we design a minimal two-level multiple-output network of AND and OR gates, one first generates all MOPI terms, MOCO terms, MOPI alterms, and MOCO alterms; then among them a subset is selected which gives the minimal cost. The detail of this design scheme will be given in the next chapters. An outline is as follows.

All MOPI terms, MOPI alterms, MOCO terms, and MOCO alterms can be generated from the basic set of MO elements, which are generated by the same scheme as the McCluskey tabular method for generation of prime implicants. All the complemented as well as uncomplemented functions are taken into consideration. From the basic set of MO elements, a sufficient set of MO elements is formed which is the collection of all MOPI terms, MOPI alterms, MOCO terms, and MOCO alterms, excluding those MOPI alterms which

satisfy the condition stated in Theorem 2.4.1 of the next section. A minimal network is designed, based on the minimal selection of the MO elements in the sufficient set of MO elements. One way of achieving this is by the integer linear programming method described in detail in Section 3.4.

2.4 Properties of MO Elements

Under the cost criterion defined previously, i.e., the minimum sum of the number of gates and the number of gate-inputs, some MOPI alterms which satisfy the condition stated in the next theorem need not be taken into consideration in designing a minimal multiple-output network.

Theorem 2.4.1: If A is a MOPI alterm with respect to some product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ (or $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$), and if $k \cdot \ell \leq 6$, then A need not be taken into consideration in the design of a minimal multiple-output network, where k is the number of literals in A and ℓ is the number of functions in the above product.

Proof: From the discussion in Section 2.3, note that if A is a MOPI alterm with respect to the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ and if the OR gate which produces the MOPI alterm A is a part of a minimal network, then the OR gate (which produces A) can be shared by f_{j_1}, f_{j_2}, ..., f_{j_ℓ}, each of which has an OR gate as its output gate; in other words, f_{j_1}, f_{j_2}, ..., f_{j_ℓ} can be written in OR-JOINT.

The case in which A is a MOPI alterm with respect to the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ will be proved in the following. The case when A is a MOPI alterm with respect to the product of functions $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$ can be similarly proved by replacing all OR gates in the former case by AND gates.

There are three possibilities for $k \cdot \ell \leq 6$.

Case 1. $k = 1$ and $\ell \leq 6$, or $\ell = 1$ and $k \leq 6$.

Case 2. $k = 2$ and $\ell = 2, 3$.

Case 3. $k = 3$ and $\ell = 2$.

According to Definition 2.2.3, k and ℓ must be ≥ 2; hence, case 1 does not exist.

In case 2, A is an alterm of 2 literals. If $\ell = 2$, then the OR gate which produces A can be shared by two output functions. The more output functions share an OR (AND) gate the more cost saving can be attained in a network. So one needs only to prove that if the use of the OR gate which is shared by two output functions does not improve the cost saving, then the MOPI alterm A needs not to be taken into consideration in designing a minimal network.

If A is shared by two output functions in a network as shown in Figure 2.8, then this alterm A has the cost of 5 which consists of 4 gate-inputs and an OR gate.

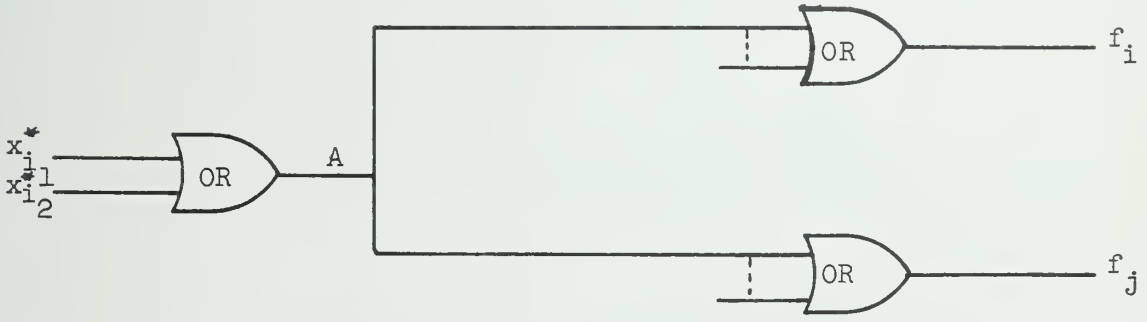


Figure 2.8

Figure 2.8 is a part of a network whose cost can be reduced if one uses individual single literals $x_{i_1}^*$, $x_{i_2}^*$ instead of the MOPI alterm A as shown in Figure 2.9,

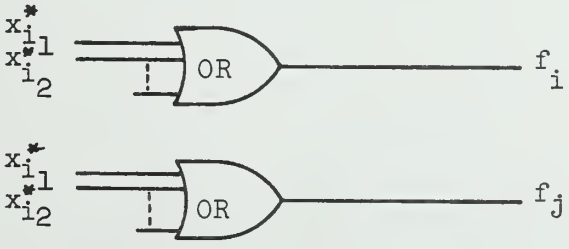


Figure 2.9

which at least reduces the total cost by 1. Thus alterm A need not be considered.

If A is shared by three output functions in a network, no reduction of cost can be made as above, but no increase of cost results in the individual

single literals of A are used instead of A. This is shown and compared in Figures 2.10 and 2.11.

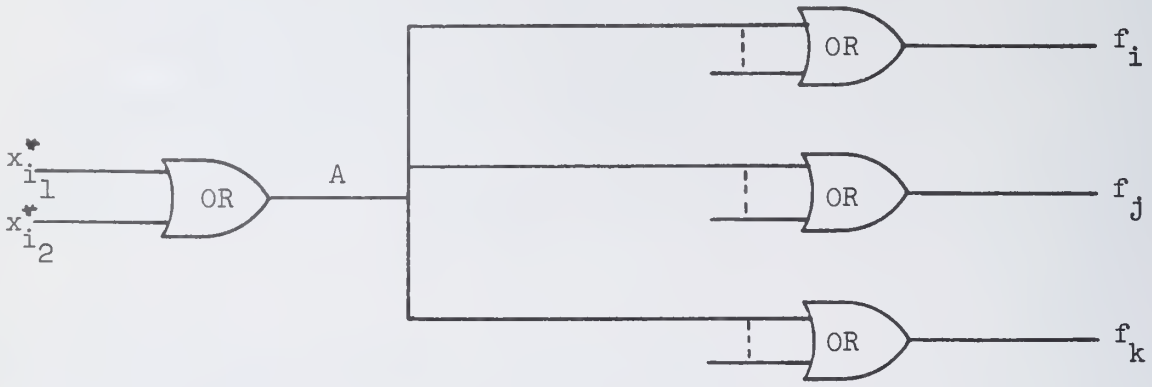


Figure 2.10

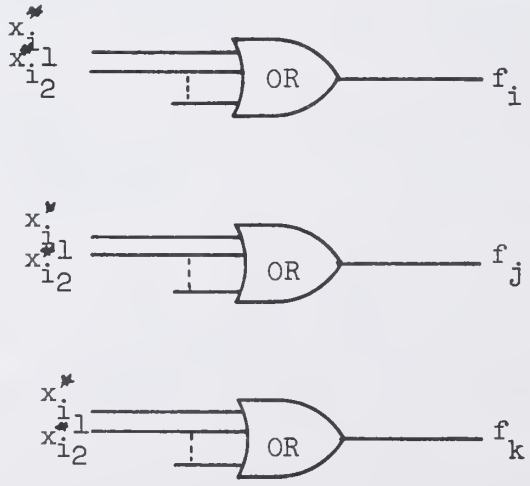


Figure 2.11

In case 3, the MOPI alterm A is of 3 literals and can be shared by 2 output functions. As in case 2, if one compares Figure 2.13 where 3 single literals $x^*_{i_1}$, $x^*_{i_2}$, $x^*_{i_3}$ of A are used instead of the MOPI alterm A with Figure 2.12 where the MOPI alterm A is used in a network, no increase of cost results.

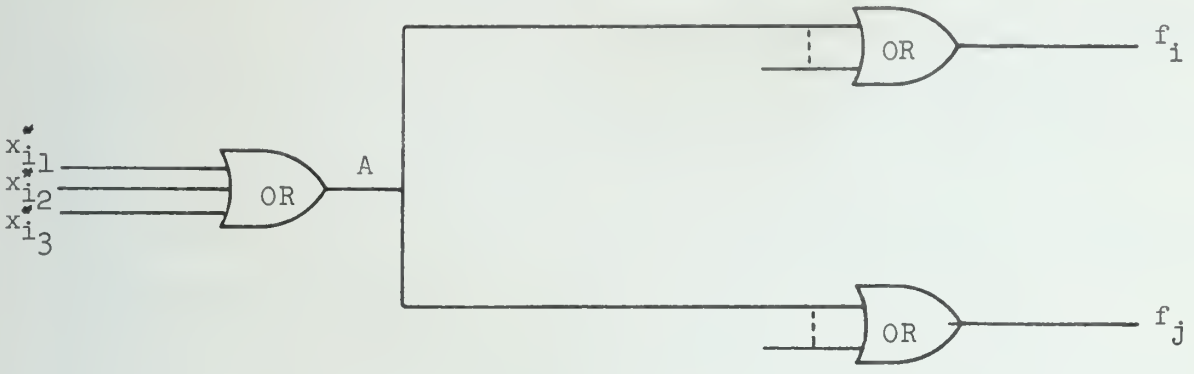


Figure 2.12

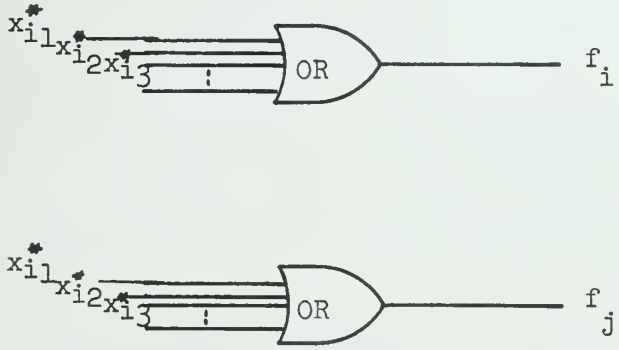


Figure 2.13

Since individual single literals of A always exist among the MOPI terms, the consideration of such A with $k \cdot \ell \leq 6$ does not give improvement of cost to a network; therefore, all those MOPI alterms need not to be taken into consideration in designing a minimal cost network.

Q.E.D.

Definition 2.4.1: The collection of all MOPI terms, MOCO terms, MOPI alterms and MOCO alterms generated from the set of output functions f_1, f_2, ..., f_m, excluding those MOPI alterms which satisfy the condition of Theorem

2.4.1 is called the sufficient set of MO elements for the set of output functions $f_1, f_2, \ldots, f_m$ and is denoted by P.

Not only every function f_i for $i \in \{1, 2, \ldots, m\}$ can have a Boolean expression which is a disjunction/conjunction of terms and alterms from the sufficient set of MO elements of $\{f_1, f_2, \ldots, f_m\}$, but also a minimal two-level AND-OR network which realizes $f_1, f_2, \ldots, f_m$ can be designed based on the sufficient set of MO elements of $f_1, f_2, \ldots, f_m$. The fact that the sufficient set of MO elements is all that need be considered in designing a minimal two-level multiple-output network is shown by the following theorem.

Theorem 2.4.2: Under the cost function defined previously (i.e., the number of gates and gate-inputs), there exists a minimal cost two-level network $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ realizing a given set of output functions $f_1, f_2, \ldots, f_m$ where $f_i = \Phi_i$ for all $i = 1, 2, \ldots, m$, such that for all i either Φ_i or $\bar{\Phi}_i$ is expressed in a disjunction of terms and/or alterms contained in the sufficient set of MO elements generated from the set of output functions $f_1, f_2, \ldots, f_m$.

(MOPI terms, MOPI alterms, MOCO terms, and MOCO alterms contained in the sufficient set of MO elements correspond to the multiple-output prime implicants in McCluskey's multiple-output prime implicant theorem.)

Proof: Since the output functions in a minimal network $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ are expressed either in OR-JOINT or AND-JOINT, the set of output functions can be grouped into two disjoint subsets I and R such that $f_i = \Phi_i$ is an OR-JOINT for $i \in I$, and $f_i = \Phi_i$ is an AND-JOINT for $i \in R$.

If Φ_i is an AND-JOINT, then $\bar{\Phi}_i$ is an OR-JOINT. Thus,

$$\Phi_i = T_{i_a} \vee \ldots \vee T_{i_b} \vee A_{i_c} \vee \ldots \vee A_{i_d} \quad \text{if } i \in I$$

and

$$\bar{\Phi}_i = T_{i_a} \vee \ldots \vee T_{i_b} \vee A_{i_c} \vee \ldots \vee A_{i_d} \quad \text{if } i \in R,$$

where $T_{i_a}, \ldots, T_{i_b}$ are terms, and $A_{i_c}, \ldots, A_{i_d}$ are alterms.

It needs to be shown that $T_{i_a}, \ldots, T_{i_b}$ are either MOPI terms or MOCO terms and that $A_{i_c}, \ldots, A_{i_d}$ are either MOPI alterms or MOCO alterms in the sufficient set of MO elements.

Assume for some $i \in I$ and $j \in \{a, \ldots, b\}$ that a term T_{i_j} is neither a MOPI term nor a MOCO term.

If T_{i_j} is not a MOPI term, then according to Definition 2.2.2, T_{i_j} is not a prime implicant of any product of functions in set I. This means that some literal(s) can be removed from T_{i_j} and the remaining term denoted by T'_{i_j} is then a prime implicant of that product of functions. Then T_{i_j} in Φ_i for all $i \in I$ can be replaced by T'_{i_j}. Let the Φ_i after the replacement of T_{i_j} by T'_{i_j} be denoted by Φ'_i, where Φ'_i still represents f. But then the network $\eta(\Phi_1, \ldots, \Phi'_i, \ldots, \Phi_m)$ costs less than $\eta(\Phi_1, \ldots, \Phi_i, \ldots, \Phi_m)$ since each T'_{i_j} in Φ'_i costs less than T_{i_j} in Φ_i. The network $\eta(\Phi_1, \ldots, \Phi'_i, \ldots, \Phi_m)$ still realizes $f_1, f_2, \ldots, f_m$. This contradicts the assumption that $\eta(\Phi_1, \ldots, \Phi_i, \ldots, \Phi_m)$ is a minimal cost network realizing $f_1, f_2, \ldots, f_m$.

If T_{i_j} is not a MOCO term, then according to Definition 2.2.4 both of the following conditions:

(1) T_{i_j} implies some product of ℓ functions in I.

(2) $\overline{T}_{i_j}$ implies some product of q complemented functions in R.

do not hold simultaneously.

The term T_{i_j} must satisfy at least one of the above two conditions because either T_{i_j} implies $f_i = \Phi_i$ if $i \in I$ or $\overline{T}_{i_j}$ implies $\overline{f}_i = \overline{\Phi}_i$ if $i \in R$. Assume that T_{i_j} satisfies (1) but not (2). If so and if T_{i_j} is not a MOPI term with respect to the product of ℓ functions in I, then some literal(s) in T_{i_j} can be removed so that the remaining term is a MOPI term which results in a network of lower cost. This is a contradiction. If T_{i_j} is a MOPI term,

we come back to the previous case.

Assume that T_{i_j} satisfies (2) but not (1), i.e., $\bar{T}_{i_j}$ implies some product of q complemented functions in R. If $\bar{T}_{i_j}$ is not a MOPI alterm with respect to this set of q complemented functions, then there exist some literal, say $x^*_{i_k}$, which can have a disjunction with $\bar{T}_{i_j}$ so that the resulting alterm $\bar{T}_{i_j} \vee x^*_{i_k}$ is a MOPI alterm with respect to the functions. It is obvious then that $x^*_{i_k}$ must imply each of the q complemented functions. In other words, the q functions must simultaneously imply $\bar{x}^*_{i_k}$. If so, $\bar{x}^*_{i_k}$ must be an input to all of the output gates (AND gates because $f_i = \Phi_i$ is an AND-JOINT for $i \in R$) of the set of q functions. This common input $\bar{x}^*_{i_k}$ for all of these output gates can be deleted after $\bar{x}^*_{i_k}$ is connected to the gate whose output is T_{i_j}. (i.e., T_{i_j} is replaced by $T'_{i_j} = \overline{\bar{T}_{i_j} \vee x^*_{i_k}} = \bar{x}^*_{i_k} T_{i_j}$.) This results in a network of lower cost which is a contradiction.

Next, assume for some $i \in I$, and $k \in \{c, \ldots, d\}$ that the alterm A_{i_k} in Φ_i is neither a MOPI alterm nor a MOCO alterm.

If A_{i_k} is not a MOPI alterm, say with respect to a set of functions $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ in the set I, then according to Definition 2.2.3 there exists at least one literal, say $x^*_{i_k}$, which can have a disjunction with A_{i_k} so that the resulting alterm $A_{i_k} \vee x^*_{i_k} = A'_{i_k}$ is a MOPI alterm with respect to $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ since A_{i_k} must satisfy the condition (1) of Definition 2.2.3 for otherwise $f_i \neq \Phi_i$, $i \in \{j_1, j_2, \ldots, f_{j_\ell}\}$. If this is so, then $x^*_{i_k}$ must also imply $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$, and accordingly $x^*_{i_k}$ is an input to the output gates (OR gates) of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$. This input $x^*_{i_k}$ can be deleted from the output gates of $f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}$ after this input is connected to the gate whose output is A_{i_k}. This results in a network of lower cost which is a contradiction.

If A_{i_k} is not a MOCO alterm, then according to Definition 2.2.5 both of the following two conditions:

(1) A_{i_k} implies some product of ℓ functions in I, and

(2) $\bar{A}_{i_k}$ implies some product of q complemented functions in R,

do not hold simultaneously.

The alterm A_{i_k} must satisfy at least one of the above two conditions because either A_{i_k} implies $f_i = \Phi_i$ if $i \in I$ or $\bar{A}_{i_k}$ implies $\bar{f}_i = \bar{\Phi}_i$ if $i \in R$.

If A_{i_k} satisfies (1) but not (2), then A_{i_k} can be replaced by a MOPI alterm which subsumes it. If A_{i_k} satisfies (2) but not (1), then $\bar{A}_{i_k}$ can be replaced by a MOPI term which is subsumed by it. In either case, a network of lower cost can be obtained since the situation that A_{i_k} is not a MOCO alterm is analogous to that of T_{i_j} being not a MOCO term after interchanging the set of ℓ functions in I and the set of q functions in R. But this is a contradiction.

The remaining case that every $T_{i_a}, \ldots, T_{i_b}$ for $i \in R$ are either MOPI terms or MOCO terms and $A_{i_c}, \ldots, A_{i_d}$ are either MOPI alterms or MOCO alterms can be proved similarly. This will not be repeated here.

Q.E.D.

The above theorem assures that it is sufficient to generate all MOPI terms, MOPI alterms, MOCO terms, MOCO alterms for designing a minimal cost two-level multiple-output network. This is where the name "sufficient" set of MO elements which contains those MOPI terms, MOPI alterms, MOCO terms, and MOCO alterms comes from. The MOPI alterms which satisfy the condition of Theorem 2.4.1 and will never give any improvement in the network cost naturally need not be included in the sufficient set of MO elements.

Properties of MO elements are to be explored in the remaining of this section. Essential MO elements will also be defined. Their properties are investigated at the end of this section.

Theorem 2.4.3: Assume that two functions f_1 and f_2 are incomparable. Then there exists neither a MOCO alterm nor a MOCO term for f_1 and f_2.

Proof: If there exists a MOCO alterm A for f_1 and f_2, then according to Definition 2.2.4, either A implies f_1 and is implied by f_2, or A implies f_2 and is implied by f_1.

In the first case, $f_1 \supseteq A \supseteq f_2$, or $f_1 \supseteq f_2$. This means f_1 and f_2 are comparable. In the second case, $f_2 \supseteq A \supseteq f_1$, or $f_2 \supseteq f_1$. This also means that f_1 and f_2 are comparable. In either case, it is a contradiction.

If there exists a MOCO term for f_1 and f_2, then the same contradiction results.

Q.E.D.

An immediate result from the proof of the above theorem is stated in the following corollary.

Corollary 2.4.1: Let f_1 and f_2 be two different functions. Then the existence of a MOCO term (or a MOCO alterm) for f_1 and f_2 implies that f_1 and f_2 are comparable.

The above theorem can be extended to two sets of functions instead of two single functions.

Corollary 2.4.2: Let $g_1 = f_{i_1} \cdot f_{i_2} \cdot f_{i_3} \cdot \ldots \cdot f_{i_k}$ and $g_2 = f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ be two products of functions. Then there exists neither a MOCO term nor a MOCO alterm for g_1 and g_2 if g_1 and g_2 are incomparable.

Henceforth g_1 and g_2 will be used to denote two different products of functions. The incomparability of g_1 and g_2 is equivalent to the conditions:

$g_1 \cdot g_2 \subset g_1$ and $g_1 \cdot g_2 \subset g_2$.

The checking of incomparability by the above two implication relations is easier.

Thus, Theorem 2.4.3 and Corollary 2.4.2 can be restated in the following theorem.

Theorem 2.4.4: If two different product of functions g_1 and g_2 have the property that

$g_1 \cdot g_2 \subset g_2$ and $g_1 \cdot g_2 \subset g_2$;

then there exists neither MOCO term nor MOCO alterm for g_1 and g_2.

For any two functions f_1 and f_2 where $f_1 \neq \bar{f}_2$, either $f_1 \cdot f_2 \neq 0$ or $\bar{f}_1 \cdot \bar{f}_2 \neq 0$ or both should be true since if $f_1 \cdot f_2 = 0$, then $\bar{f}_1 \cdot \bar{f}_2 \neq 0$ and if $\bar{f}_1 \cdot \bar{f}_2 = 0$, then $f_1 \cdot f_2 \neq 0$. Thus, the following theorem can be derived.

Theorem 2.4.5: Let f_1 and f_2 be two functions, and $f_1 \neq \bar{f}_2$. Then there exists at least one MOPI term for f_1 and f_2.

Proof: Assume that there exists no MOPI term for f_1 and f_2, then according to Definition 2.2.2 there is no term T which implies either $f_1 \cdot f_2$ or $\bar{f}_1 \cdot \bar{f}_2$. But this is possible only when $f_1 \cdot f_2 = 0$ and $\bar{f}_1 \cdot \bar{f}_2 = 0$ which does not occur for f_1 and f_2, where $f_1 \neq \bar{f}_2$.

Q.E.D.

The above theorem holds also for two sets of products of functions g_1 and g_2 instead of f_1 and f_2.

The existence of MO elements implies a certain implication relation between/among functions as the following theorem shows.

Lemma 2.4.3: If a term (or an alterm) implies both f_1 and f_2 and is essentially neither of them, then the following implication relations hold:

1. $f_1 \cdot \bar{f}_2 \subset \bar{f}_2$ and $f_1 \cdot \bar{f}_2 \subset f_1$
2. $\bar{f}_1 \cdot f_2 \subset \bar{f}_1$ and $\bar{f}_1 \cdot f_2 \subset f_2$

Proof: Since f_1 and f_2 are both implied by a term T, f_1 and f_2 can be written as

$f_1 = T \vee \ldots$

$f_2 = T \vee \ldots$

where ... indicates some other terms and alterms.

Clearly, $f_1 \cdot \bar{f}_2 \subset \bar{T} \subseteq \bar{f}_2$ and $f_1 \cdot \bar{f}_2 \subset f_1$ by direct conjunction of f_1 and $\bar{f}_2$. Similarly, if the conjunction of $\bar{f}_1$ and f_2 is formed then

$\bar{f}_1 \cdot f_2 \subset \bar{f}_1 \cdot f_2 \subset f_2$.

The proof for the case of an alterm is similar.

Q.E.D.

When the term and alterm in the above Lemma 2.4.3 are actually MOPI term and MOPI alterm, respectively, the following theorem follows.

Theorem 2.4.6: Let f_1 and f_2 be two output functions. Then the existence of a MOPI term (or a MOPI alterm) with respect to f_1 and f_2 implies the existence of the following implication relations:

(1) $f_1 \cdot \bar{f}_2 \subset \bar{f}_2$ and $f_1 \cdot \bar{f}_2 \subset f_1$

(2) $\bar{f}_1 \cdot f_2 \subset \bar{f}_1$ and $\bar{f}_1 \cdot f_2 \subset f_2$.

Theorem 2.4.7: Let A be a MOCO alterm for a set of output functions. Then there exists at least one MOPI term which is subsumed by $\bar{A}$ for the same set of output functions.

Proof: Assume that A is a MOCO alterm for the set of output functions $(f_1, f_2, \ldots, f_m)$. Then according to Definition 2.2.5 there exist two non-empty disjoint subsets of functions $(f_{j_1}, f_{j_2}, \ldots, f_{j_\ell})$ and $(f_{p_1}, f_{p_2}, \ldots, f_{p_q})$ of $(f_1, f_2, \ldots, f_m)$ such that

A implies $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$,

and $\bar{A}$ implies $\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$.

There always exists a prime implicant of the product $\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$ which is subsumed by $\bar{A}$. But according to Definition 2.2.2 this prime implicant is a MOPI term.

Q.E.D.

An immediate result followed from the proof of the above theorem is stated in the next corollary.

Corollary 2.4.2: The existence of a MOCO alterm implies the existence of a MOPI alterm which subsumes the MOCO alterm for a given set of output functions.

Proof: There always exists an alterm which subsumes A and is a MOPI alterm

with respect to $f_{j_1}, f_{j_2}, \ldots, f_{\ell}$.

Q.E.D.

For a MOCO term a similar result can be obtained which is stated in the next theorem and corollary. The proofs are similar to the above. Therefore, they are omitted.

Theorem 2.4.8: Let T be a MOCO term for a set of output functions; then there exists at least one MOPI alterm which subsumes $\overline{T}$ for the same set of output functions.

Corollary 2.4.3: The existence of a MOCO term implies the existence of a MOPI term which is subsumed by the MOCO term for a given set of output functions.

The distinguished fundamental product and essential MO element are defined, and related properties are presented in the following.

Definition 2.4.2: Let $F = \{f_1, f_2, \ldots, f_m\}$ be a set of output functions and P be the sufficient set of MO elements of F. A fundamental product p is called a distinguished fundamental product if p implies only one MO element of P. The MO element implied by a fundamental product is called essential MO element, or simply essential.

Lemma 2.4.4: Neither MOPI alterm nor MOCO alterm in P is essential.

Proof: If a MOPI alterm is essential, then there exists some fundamental product p which implies only this MOPI alterm denoted by $x^*_{i_1} \vee x^*_{i_2} \vee \ldots \vee x^*_{i_k}$. This means that some literal, say $x^*_{i_j}$ $j \in \{1, 2, \ldots, k\}$, is implied by p. But $x^*_{i_j}$ is also a MOPI term in P from the definition of a MOPI term. Since $x^*_{i_1} \vee \ldots \vee x^*_{i_k}$ implies some product of functions because it is a MOPI alterm, $x^*_{i_j}$ also implies that product of functions. In other words p implies $x^*_{i_1} \vee \ldots \vee x^*_{i_k}$ and also $x^*_{i_j}$. This contradicts the assumption that the MOPI alterm is essential.

The proof for a MOCO alterm is similar to that of a MOPI alterm.

Q.E.D.

The above lemma shows that only a MOPI term or a MOCO term can be an essential MO element. The essential MO term is abbreviated as EMO term to mean an essential MO element.

Lemma 2.4.5: There exist no two EMO terms in P of F in which one is subsumed by the other.

Proof: Since if a distinguished fundamental product implies an EMO term which subsumes another EMO term, then the latter is also implied by the same distinguished fundamental product. This is a contradiction.

Q.E.D.

A property of EMO term is shown in the following.

Theorem 2.4.9: Let $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_m)$ be a minimal network realizing $F = \{f_1, f_2, \ldots, f_m\}$ and $f_i = \Phi_i$ for $i \in \{1, 2, \ldots, m\}$. If $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_m)$ is synthesized based on the sufficient set of MO elements of F, then all EMO terms which are implied by distinguished fundamental products of each f_i (or $\bar{f}_i$) must be contained in either Φ_i or $\bar{\Phi}_i$.

Proof: Assume that some EMO term T which is implied by a distinguished fundamental product p of f_i (or $\bar{f}_i$) is not included in Φ_i and $\bar{\Phi}_i$. But this is impossible since $f_i = \Phi_i$ and $\bar{f}_i = \bar{\Phi}_i$ and p must be either in f_i or $\bar{f}_i$. In other words, p must imply either Φ_i or $\bar{\Phi}_i$.

Q.E.D.

In the following, the residual function of an output function is defined. The use of residual function with respect to a minimal network realization will also be explored.

Definition 2.4.3: Let the disjunction of all distinguished fundamental products of f_i be denoted by $h\ (f_i)$, and the disjunction of all distinguished fundamental products of $\bar{f}_i$ be denoted by $h\ (\bar{f}_i)$. Then the

residual function $r(f_i)$ of f_i if defined such that

$f_i = r(f_i) \vee h(f_i)$, where $r(f_i) \wedge h(f_i) = \emptyset$.

The residual function $r(\bar{f}_i)$ of $\bar{f}_i$ is similarly defined such that

$\bar{f}_i = r(\bar{f}_i) \vee h(\bar{f}_i)$, where $r(\bar{f}_i) \wedge h(\bar{f}_i) = \emptyset$.

Obviously, $r(f_i)$ and $r(\bar{f}_i)$ are residual functions of f_i and $\bar{f}_i$, respectively, if and only if $r(f_i)$ and $r(\bar{f}_i)$ contains all non-distinguished fundamental products of f_i and $\bar{f}_i$, respectively.

In the definition of a sufficient set of MO elements P for $F = \{f_1, f_2, \ldots, f_m\}$, it is implicitly implied that the sufficient set of MO elements P is formed with respect to the set of functions $f_1, f_2, \ldots, f_m$ and their complements $\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_m$. Since $r(f_i)$ and $r(\bar{f}_i)$ are the residual functions of f_i and $\bar{f}_i$, respectively, the set of functions $r(f_1), r(f_2), \ldots, r(f_m)$ and $r(\bar{f}_1), r(\bar{f}_2), \ldots, r(\bar{f}_m)$ is termed as the set of residual functions of F and is denoted by $R^{\#}$. The sufficient set of MO elements which is formed with respect to $R^{\#}$, i.e., $r(f_1), r(f_2), \ldots, r(f_m)$ and $r(\bar{f}_1), r(\bar{f}_2), \ldots, r(\bar{f}_m)$, will be termed as the sufficient set of MO elements for the set of residual functions $R^{\#}$ of F, and is denoted by $P^{\#}$.

The sufficient set of MO elements for F and the sufficient set of MO elements for $R^{\#}$ have the following relationship.

Theorem 2.4.10: Let $F = \{f_1, f_2, \ldots, f_m\}$ be a set of output functions and E be the set of all essential MO elements of F Then $P = P^{\#} \cup E$ is a sufficient set of MO elements for F if and only if $P^{\#}$ is a sufficient set of MO elements for $R^{\#}$ where $R^{\#}$ is the set of residual functions of F.

Proof: The "if" part. Assume that $P^{\#}$ is a sufficient set of MO elements for $R^{\#}$. Then according to Theorem 2.4.2 every residual function $r(f_i)$ $(r(\bar{f}_i))$ in $R^{\#}$ has a Boolean expression which is a disjunction of MO elements in $P^{\#}$. But $f_i = r(f_i) \vee h(f_i)$ (also $\bar{f}_i = r(f_i) \vee h(f_i)$).

The disjunction of the Boolean expression of r (f_i) (also r ($\bar{f}_i$)) and the distinguished fundamental products of f_i (also $\bar{f}_i$) i.e., h (f_i) (h ($\bar{f}_i$)) is a Boolean expression of f_i ($\bar{f}_i$). But every distinguished fundamental product of f_i ($\bar{f}_i$) implies one MO element in E by the definition of the set E. Hence, the Boolean expression of f_i ($\bar{f}_i$) can be replaced by the disjunction of the Boolean expression of r (f_i) (r ($\bar{f}_i$)) and those MO elements each of which is implied by a distinguished fundamental product of f_i ($\bar{f}_i$), i.e., the fundamental products of h (f_i) (h ($\bar{f}_i$)). This is true for i = 1, 2, ..., m. By Theorem 2.4.2 the set $P^{\#}$ U E is thus a sufficient set of MO elements for the set of functions F = {f_1, f_2, ..., f_m}.

The "only if" part. Assume that P is a sufficient set of MO elements for F and $P^{\#}$ is <u>not</u> a sufficient set of MO elements for $R^{\#}$. Then there exists at least one fundamental product p of some residual function in $R^{\#}$ which does not imply any member of $P^{\#}$. This fundamental product p cannot be a distinguished fundamental product of f_i (or $\bar{f}_i$) of any i $\in$ {1, 2, ..., m} since by definition $R^{\#}$ contains no distinguished fundamental product of f_i and $\bar{f}_i$. But p is a fundamental product of f_i (or $\bar{f}_i$). It concludes that $P^{\#}$ U E cannot be a sufficient set of F because there is a fundamental product p of some function f_i ($\bar{f}_i$) which does not imply any member of $P^{\#}$ U E. Consequently, not all functions $f_i \in F$ (or $\bar{f}_i$) can have a Boolean expression by forming a disjunction of some MO elements in the set $P^{\#}$ U E. This is contradictory to the assumption.

Q.E.D.

Let e (f_i) be the disjunction of all essential MO elements of f_i and e ($\bar{f}_i$) be the disjunction of all essential MO elements of $\bar{f}_i$. Note that e (f_i) is implied by each of the distinguished fundamental product of f_i and e ($\bar{f}_i$) is implied by each of the distinguished fundamental product of $\bar{f}_i$.

The relationship between two minimal networks, one realizing the set of output functions $f_1, f_2, \ldots, f_m$ and the other realizing the set of residual functions of $f_1, f_2, \ldots, f_m$, is stated in the following theorem.

Theorem 2.4.11: Let $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ and $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ be two multiple-output two-level AND-OR networks realizing $F = \{f_1, f_2, \ldots, f_m\}$ and $\{r(f_1), r(f_2), \ldots, r(f_m)\}$, respectively, where $f_i = \Phi_i$ and $r(f_i) = \Phi'_i$ for all $i \in \{1, 2, \ldots, m\}$. Assume that $r(f_i)$ is the residual function of f_i and $\Phi_i = \Phi'_i \vee e(f_i)$ (or $\bar{\Phi}_i = \bar{\Phi}'_i \vee e(\bar{f}_i)$ for some i) where $e(f_i)$ and $e(\bar{f}_i)$ are the disjunction of all essential MO elements of f_i and $\bar{f}_i$, respectively. Then $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ has the minimum cost if $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ has the minimum cost.

Proof: Assume that $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ has the minimum cost and $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ has not. Then either some term or alterm in some Φ'_i or $\bar{\Phi}'_i$ can be replaced or deleted and results in a network of lower cost which still realizes $\{r(f_1), r(f_2), \ldots, r(f_m)\}$. If this is possible, then it is also possible to do the same replacement or deletion in Φ_i since $\Phi_i = \Phi'_i \vee e(f_i)$ (or $\bar{\Phi}_i$ for $\bar{\Phi}'_i \vee e(\bar{f}_i)$ for some i). This results in a network of lower cost which realizes F. But this is a contradiction since $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ is a network of minimum cost.

Q.E.D.

The converse of the above theorem is not necessarily true except the case in which $e(f_i)$ and/or $e(\bar{f}_i)$ is a single essential MO element, because if $e(f_i)$ (or $e(\bar{f}_i)$) has more than one essential MO element, then their disjunction may be implied by some other term or alterm in Φ'_i (or $\bar{\Phi}'_i$) for $\Phi_i = \Phi'_i \vee e(f_i)$ (or $\bar{\Phi}_i = \bar{\Phi}'_i \vee e(\bar{f}_i)$). This implies that the network $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ is not a minimum cost network even though $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ is a network of minimum cost. Thus the above theorem can be strengthened to a necessary and sufficient condition when each $e(f_i)$ and $e(f_i)$ consists

of only one essential MO element.

Theorem 2.4.12: Let $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ and $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ be two two-level AND-OR networks realizing $F = \{f_1, f_2, \ldots, f_m\}$ and $\{r(f_1), r(f_2), \ldots, r(f_m)\}$, respectively, where $f_i = \Phi_i$ and $r(f_i) = \Phi'_i$ for all $i \in \{1, 2, \ldots, m\}$. Assume that $r(f_i)$ is the residual function of f_i and $\Phi_i = \Phi'_i \vee e(f_i)$ (or $\bar{\Phi}_i = \bar{\Phi}'_i \vee e(\bar{f}_i)$ for some i) and that each $e(f_i)$ (or $e(\bar{f}_i)$) consists of a single essential MO element of f_i (or $\bar{f}_i$); then the network $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ has the minimum cost if and only if the network $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ has the minimum cost.

Proof: The "only if" part is proved in Theorem 2.4.11. The "if" part is proved in the following.

Assume that $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ is a network of minimum cost and $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ is not. Then there exists at least one term or alterm in Φ_i (or $\bar{\Phi}_i$ for some i) for $i \in \{1, 2, \ldots, m\}$ which can be either replaced or deleted without violating $f_i = \Phi_i$ condition such that a network of lower cost can be achieved. If this is possible, then this term or alterm must be in Φ'_i of the expression Φ_i (or $\bar{\Phi}'_i$ of $\bar{\Phi}_i$) since otherwise $e(f_i)$ (or $e(\bar{f}_i)$) can not be essential. If this is so, then after such replacement or deletion $r(f_i) = \Phi'_i$ still holds and results in a network of lower cost. But this is a contradiction.

Q.E.D.

The above theorem provides a means to reduce a minimization problem of multiple-output functions to a smaller problem where only the residual functions are taken into consideration whenever the distinguished fundamental products and the EMO terms are found.

The uniqueness of a minimal network of multiple-output under a certain special condition is stated in the following.

Theorem 2.4.13: $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_n)$ is a unique minimal cost network realizing $F = \{f_1, f_2, \ldots, f_m\}$ where $f_i = \Phi_i$ for all $i = 1, 2, \ldots, m$ if all the residual functions of f_i are the zero function.

3. THE ALGORITHM

3.1 Introduction

The algorithm is divided into four parts. 1. The generation of the basic set of MO elements. 2. The generation of the sufficient set of MO elements. 3. The selection of a minimal covering. 4. The drawing of the minimal network.

The first part of this algorithm is an extension of McCluskey's technique for finding prime implicants by considering the complemented functions $\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_m$ as well as the uncomplemented ones, $f_1, f_2, \ldots, f_m$. The basic set of MO elements is found if all the multiple-output prime implicants of $f_1, f_2, \ldots, f_m, \bar{f}_1, \bar{f}_2, \ldots, \bar{f}_m$ are found. The second part of this algorithm is to detect the possible existence of MOCO terms, MOPI alterms and MOCO alterms from the basic set of MO elements obtained in the first part of this algorithm. The sufficient set of MO elements is formed by collecting all of the MOPI terms, MOCO terms, MOCO alterms, and MOPI alterms, excluding those MOPI alterms satisfying the condition in Theorem 2.4.1.

The third part of this algorithm is to select a minimal covering set from the sufficient set of MO elements. This is done by the integer linear programming method whose objective function is the cost function of the network to be designed. The optimal solution of the integer linear programming problem guarantees that the network to be designed has the minimum cost.

The last part of this algorithm is to write down the Boolean expressions (or the Boolean forms) of each output function and to draw the logic diagram of the two-level multiple-output network realizing the given set of output functions.

3.2 The Generation of the Basic Set of MO Elements

As has been mentioned before, McCluskey's tabular method for finding the multiple-output prime implicants will be extended to generate the basic set of MO elements.

Readers are assumed to be familiar with the McCluskey tabular method for finding the multiple-output prime implicants. In his method, he uses binary characters each of which consists of two parts--the identifier, which is to identify the fundamental product, and the tag, which specifies which of the output functions $f_1, f_2, \ldots, f_m$ include the fundamental product specified by the identifier portion of the character. The binary character in McCluskey's method is denoted as $\epsilon = (\epsilon_1; \epsilon'_2)$, where ϵ_1 and ϵ_2 are the identifier and tag, respectively.

Since the basic set of MO elements is simply the collection of all MOPI terms, the method for generating the MOPI terms will be described. The method is an extension of McCluskey's tabular method in which the binary character ϵ^* is used. ϵ^* consists of ϵ_1 and ϵ^*_2 where ϵ_1 is the same as the above and the ϵ^*_2 part is to specify which of the functions $f_1, f_2, \ldots, f_m$ and $\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_m$ include the fundamental product specified by the ϵ_1 part. Actually, the ϵ^*_2 part can be written as $(\epsilon'_2, \epsilon''_2)$, where ϵ'_2 is the tag part with respect to $f_1, f_2, \ldots, f_m$, and ϵ''_2 is the new tag part with respect to the complemented functions $\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_m$. Thus, $\epsilon^* = (\epsilon_1; \epsilon^*_2) = (\epsilon_1; \epsilon'_2, \epsilon''_2)$.

If a fundamental product is included in f_i, this fundamental product will not be included in $\bar{f}_i$ and vice versa. The ϵ''_2 is formed by entering - or 0 in its m positions whenever the corresponding position of ϵ'_2 is 0 or -, respectively. The symbol 0 in the i-th position of ϵ'_2 indicates that the fundamental product ϵ_1 is not included in an output function f_i and the symbol - indicates that the fundamental product ϵ_1 is included in an output function. For example, the fundamental product $\bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 x_5$ in the

Table 3.2 has a binary character ϵ^* = (0 0 0 1 1; - 0 0, 0 - -) with respect to f_1, f_2, and f_3. This fundamental product $\bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 x_5$ is included in f_1, $\bar{f}_2$, and $\bar{f}_3$ but not in f_2, f_3 and $\bar{f}_1$.

McCluskey's tabular method for finding the multiple-output prime implicants is well known. So its detailed procedure will not be repeated here. Readers are referred to his book, Introduction to the Theory of Switching Circuits (22).

The procedure for generating the basic set of MO elements for a given set of output functions f_1, f_2, ..., f_m are as follows.

1. Write down the truth tables for f_1, f_2, ..., f_m.
2. Form binary character ϵ^* for each fundamental product and list in the order according to the number of 1's in the identifiers.
3. Apply McCluskey's tabular method, i.e., combine every two rows which differ in only one component in ϵ_1 parts, for all binary characters ϵ *'s formed in step 2.
4. At the termination of the above step 3, the remaining identifier parts of the binary characters ϵ^*'s are converted into terms. Each term is simply a product of literals x_i, if the i-th component of the identifier part is 1, and $\bar{x}_i$ if the i-th component of the identifier part is 0 for i = 1, 2, ..., m. The collection of all such terms along with their associated ϵ_2^* parts is the basic set of MO elements for f_1, f_2, ..., f_m.

Let us illustrate the above procedure in the following. The basic set of MO elements is shown in Table 3.3.

Example: Three output functions f_1, f_2, and f_3 are given:

$$f_1 (x_1, x_2, x_3, x_4, x_5) = \Sigma (0, 2, 5, \ldots, 31)$$

$$f_2 (x_1, x_2, x_3, x_4, x_5) = \Sigma (1, 4, \ldots, 31)$$

$$f_3 (x_1, x_2, x_3, x_4, x_5) = \Sigma (11, 19, 27.)$$

Step 1. The truth table for f_1, f_2, f_3 is formed in Table 3.1.

Step 2. The binary character ϵ^* for each fundamental product is formed and listed in Table 3.2 according to the order of the number 1's in the identifier ϵ_1 part.

Step 3. Apply McCluskey's tabular method to ϵ^* s in Table 3.2.

Step 4. Write down each term corresponding to each binary character remaining in Step 3. List all terms along with their associated ϵ_2^*'s. The basic set of MO elements is obtained which is found in Table 3.3.

Table 3.1 The Truth Table.

	Fundamental Product					Functions		
	x_1	x_2	x_3	x_4	x_5	f_1	f_2	f_3
0	0	0	0	0	0	1	0	0
1	0	0	0	0	1	0	1	0
2	0	0	0	1	0	0	0	0
3	0	0	0	1	1	1	0	0
4	0	0	1	0	0	1	1	0
5	0	0	1	0	1	1	1	0
6	0	0	1	1	0	1	1	0
7	0	0	1	1	1	1	1	0
8	0	1	0	0	0	1	1	0
9	0	1	0	0	1	1	1	0
10	0	1	0	1	0	1	1	0
11	0	1	0	1	1	1	1	1
12	0	1	1	0	0	1	1	0
13	0	1	1	0	1	1	1	0
14	0	1	1	1	0	1	1	0
15	0	1	1	1	1	1	1	0

Table 3.1. (Continued)

16	1	0	0	0	0	1	1	0
17	1	0	0	0	1	1	1	0
18	1	0	0	1	0	1	1	0
19	1	0	0	1	1	1	1	1
20	1	0	1	0	0	1	1	0
21	1	0	1	0	1	1	1	0
22	1	0	1	1	0	1	1	0
23	1	0	1	1	1	1	1	0
24	1	1	0	0	0	1	1	0
25	1	1	0	0	1	1	1	0
26	1	1	0	1	0	1	1	0
27	1	1	0	1	1	1	1	1
28	1	1	1	0	0	1	1	0
29	1	1	1	0	1	1	1	0
30	1	1	1	1	0	1	1	0
31	1	1	1	1	1	1	1	0

Table 3.2. The Binary Characters, ϵ^*.

	ϵ_1					ϵ'_2			ϵ''_2		
Decimal Number	x_1	x_2	x_3	x_4	x_5	f_1	f_2	f_3	$\bar{f}_1$	$\bar{f}_2$	$\bar{f}_3$
0	0	0	0	0	0	-	0	0	0	-	-
1	0	0	0	0	1	0	-	0	-	0	-
2	0	0	0	1	0	0	0	0	-	-	-
4	0	0	1	0	0	-	-	0	0	0	-
8	0	1	0	0	0	-	-	0	0	0	-
16	1	0	0	0	0	-	-	0	0	0	-
3	0	0	0	1	1	-	0	0	0	-	-
5	0	0	1	0	1	-	-	0	0	0	-
9	0	1	0	0	1	-	-	0	0	0	-
17	1	0	0	0	1	-	-	0	0	0	-
6	0	0	1	1	0	-	-	0	0	0	-
10	0	1	0	1	0	-	-	0	0	0	-
18	1	0	0	1	0	-	-	0	0	0	-
12	0	1	1	0	0	-	-	0	0	0	-
20	1	0	1	0	0	-	-	0	0	0	-
24	1	1	0	0	0	-	-	0	0	0	-
7	0	0	1	1	1	-	-	0	0	0	-
11	0	1	0	1	1	-	-	-	0	0	0
19	1	0	0	1	1	-	-	-	0	0	0
13	0	1	1	0	1	-	-	0	0	0	-
21	1	0	1	0	1	-	-	0	0	0	-
25	1	1	0	0	1	-	-	0	0	0	-
14	0	1	1	1	0	-	-	0	0	0	-
22	1	0	1	1	0	-	-	0	0	0	-
26	1	1	0	1	0	-	-	0	0	0	-

Table 3.2. (Continued)

28	1	1	1	0	0	-	-	0	0	0	-
15	0	1	1	1	1	-	-	0	0	0	-
23	1	0	1	1	1	-	-	0	0	0	-
27	1	1	0	1	1	-	-	-	0	0	0
29	1	1	1	0	1	-	-	0	0	0	-
30	1	1	1	1	0	-	-	0	0	0	-
31	1	1	1	1	1	-	-	0	0	0	-

Table 3.3. The Basic Set of MO Elements

Fundamental Products	MOPI terms	ϵ_1	ϵ'_2	ϵ''_2
(16, ..., 31)	x_1	1 - - - -	- - 0	0 0 0
(8, ..., 15, 24, ..., 31)	x_2	- 1 - - -	- - 0	0 0 0
(4, ..., 7, 12, ..., 15, 20, ..., 23, 28, ..., 31)	x_3	- - 1 - -	- - 0	0 0 -
(0, 2, 4, 8, 10, 12, 14, 16, 18, 20, 20, 24, 26, 28, 30)	$\bar{x}_4$	- - - 0 -	0 0 0	0 0 -
(0, 1, 4, 5, 8, 9, 12, 13, 16, 17, 20, 21, 24, 25, 28, 29)	$\bar{x}_5$	- - - - 0	0 0 0	0 0 -
(3, 7, 11, 15, 19, 23, 29, 31)	x_4x_5	- - - 1 1	- 0 0	0 0 -
(16, 17, 20, 21, 24, 25, 28, 29)	$x_1\bar{x}_4$	1 - - 0 1	- - 0	0 0 -
(8, 9, 12, 13, 24, 25, 28, 29)	$x_2\bar{x}_4$	- 1 - 0 -	- - 0	0 0 -
(1, 5, 9, 13, 17, 21, 25, 29)	$\bar{x}_4x_5$	- - - 0 1	0 - 0	0 0 -
(3, 4, 8, 12, 16, 20, 24, 28)	$\bar{x}_4\bar{x}_5$	- - - 0 0	- 0 0	0 0 -
(0, 1, 2, 3, 4, 5, 6, 7)	$\bar{x}_1\bar{x}_2$	0 0 - - -	0 0 0	0 0 -
(6, 7, 14, 15)	$\bar{x}_1x_3x_4$	0 - 1 1 -	- - 0	0 0 -
(3, 7, 19, 23)	$\bar{x}_2x_4x_5$	- 0 - 1 1	- 0 0	0 0 0
(3, 11, 17, 27)	$\bar{x}_3x_4x_5$	- - 0 1 1	- 0 0	0 0 0
(19, 27)	$x_1\bar{x}_3x_4x_5$	1 - 0 1 1	- - -	0 0 0
(11, 27)	$x_2\bar{x}_3x_4x_5$	- 1 0 1 1	- - -	0 0 0
(3, 7)	$\bar{x}_1\bar{x}_2x_4x_5$	0 0 - 1 1	- 0 0	0 0 -
(2, 3)	$\bar{x}_1\bar{x}_2\bar{x}_3x_4$	0 0 0 1 -	0 0 0	0 - -
(0, 2)	$\bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_5$	0 0 0 - 0	0 0 0	0 - -

3.3 The Generation of the Sufficient Set of MO Elements.

After the basic set of MO elements is found, the MOPI alterms, MOCO alterms and MOCO terms can be found from the basic set of MO elements. Three procedures for detecting the existence of MOPI alterms, MOCO alterms and MOCO terms are given in the following. The procedure for the generation of MOCO terms is similar to the procedure for the generation of MOCO alterms.

A. Procedure for the generation of MOPI alterms.

1. Form the disjunction of single literal MOPI terms in the basic set of MO elements whose ϵ'_2 parts are the same. The set of all disjunctions made for different ϵ'_2's is denoted by S_1.
2. Form the disjunction of single literal MOPI terms in the basic set of MO elements whose ϵ''_2 parts are the same. The set of all disjunctions made for different ϵ''_2's is denoted by S_2.
3. Since every alterm in S_1 implies the product of functions which have dashes in the components of ϵ'_2 part, every alterm in S_2 implies the product of the complement of functions which have dashes in the components of ϵ''_2 part, and every alterm in S_1 and S_2 has the maximum number of literals which imply a product of functions, the collection of S_1 and S_2 is then the set of MOPI alterms.
4. Write down the MOPI alterms and their associated ϵ^*_2 parts, where $\epsilon^*_2 = (\epsilon'_2, \epsilon''_2)$. The ϵ^*_2 is formed according to the following process:

   Any ϵ'_2 part of the literal in a MOPI alterm is taken as the ϵ'_2 part of a MOPI alterm in S_1, and the ϵ''_2 part of a MOPI alterm in S_1 is formed according to

   (a) a component of ϵ''_2 is 0 if the corresponding components of ϵ''_2 of any literal in the MOPI alterm is 0.

(b) a component of ϵ''_2 is - if the corresponding components of ϵ''_2 of all literals in the MOPI alterm is -.

The ϵ^*_2 of a MOPI alterm in S_2 is formed in the same way as ϵ^*_2 of a MOPI alterm in S_1 except the roles of ϵ'_2 and ϵ''_2 interchanged.

B. Procedure for the generation of MOCO alterms.

1. Let $A_1, A_2, \ldots, A_I$ be the alterms in the S_1 set of MOPI alterms. Assume that A_i has $\ell(i)$ literals. The ϵ'_2 part of A_i is denoted as $\epsilon'_2(i)$.

   Set indices i = 1 and r = 1, where r-1 denotes the number of literals which have been removed from A_i during this procedure.

2. Obtain the term $T_i = \bar{A}_i$ for an alterm A_i.

3. Check whether or not T_i subsumes/is a MOPI term in the basic set of MO elements whose ϵ''_2 part has at least one dash (-) which is not in the corresponding component of $\epsilon'_2(i)$ of A_i. (This condition on ϵ''_2 part is due to disjointedness of functions in Definition 2.2.4.)

   If yes, go to 5. Otherwise, go to 4.

4. Set i = i + 1, and reset r = 1.

   If $i \leq I$ then go to 2; otherwise, stop.

5. Record the pair of T_i and $\bar{T}_i$ and the associated $\epsilon'_2(i)$ and ϵ''_2 part of the MOPI term just mentioned. ($\bar{T}_i$ is A_i in the beginning but becomes different from A_i after step 6.)

6. Generate new T_i's by removing one literal from T_i for each new T_i.

   Set r = r + 1. If $\ell(i) - r > 1$, then go to 3; otherwise, go to 4.

Since every $\bar{T}_i$ which is an alterm implies the product of functions which have dashes in the components of $\epsilon'_2(i)$ part ($\bar{T}_i$ implies because A_i is chosen according to procedure A. Note that $\bar{T}_i$ may eventually contain fewer literals than A_i in step 1. So does T_i.) implies the product of the

complements of functions which have dashes in the components of ϵ''_2 part of this T_i, the collection of all $\bar{T}_i$'s recorded in step 5 are then the MOCO alterms.

C. Procedure for the generation of MOCO terms.

1. Let $A_1, A_2, \ldots, A_I$ be the set of alterms in S_2 of the MOPI alterms. A_i has $\ell(i)$ literals. The ϵ''_2 part of A_i is denoted as $\epsilon''_2(i)$. Set the indices i = 1 and r = 1, where r-1 denotes the number of literals removed from A_i during the following steps.
2. Obtain the term $T_i = \bar{A}_i$, for an alterm A_i.
3. Check whether or not T_i is a MOPI term/or subsumes a MOPI term in the basic set of MO elements whose ϵ'_2 has at least one dash (-) which is not in the corresponding position of $\epsilon''_2(i)$
   If yes, go to 5; otherwise, go to 4.
4. Set i = i + 1 and reset r = 1.
   If $i \leq I$, go to 2; otherwise, stop.
5. Record this pair of T_i and $\bar{T}_i$ along with the associated $\epsilon''_2(i)$ and ϵ'_2 parts of the MOPI term just matched.
6. Generate new T_i's by removing one literal from T_i for each new T_i.
   If $\ell(i) - r > 1$, then go to 3; otherwise go to 4.

At the termination of above six steps, the collection of all T_i's recorded in step 5 is the set of all MOCO terms. This is because every T_i implies the product of functions which have dashes in the components of ϵ'_2 part of T_i and $\bar{T}_i$ implies the product of the complement of functions which have dashes in the component of $\epsilon''_2(i)$ of T_i.

Next exclude MOPI alterms whose $k \cdot \ell \leq 6$ as stated in Theorem 2.4.1, where k = the number of literals in the MOPI alterms and ℓ = the number of dashes in its associated ϵ'_2 part or ϵ''_2 part whichever is bigger. Then

Then we have the sufficient set of MO elements which is the collection of all MOPI terms, MOCO terms, MOPI alterms, and MOCO alterms.

In order to illustrate the above procedures, an example is given which is shown in Table 3.4.

Table 3.4. Example

From the basic set of MO elements shown in Table 3.3 obtained from the previous example, the following MOPI alterms, MOCO terms and MOCO alterms are obtained after the procedures A, B and C are applied (the numericals correspond to the step in each procedure.)

Applying procedure A to check the existence of MOPI alterms, one has

1. alterm $(x_1 \vee x_2 \vee x_3)$ with $\epsilon'_2 = (- - 0)$ in S_1, and
2. alterm $(x_3 \vee \bar{x}_4 \vee \bar{x}_5)$ with $\epsilon''_2 = (0\ 0\ -)$ in S_2.
3. The collection of alterms in S_1 and S_2 is the set of MOPI alterms which are:

$$(x_1 \vee x_2 \vee x_3) \qquad \epsilon^*_2 = (- - 0,\ 0\ 0\ 0)$$
$$(x_3 \vee \bar{x}_4 \vee \bar{x}_5) \qquad \epsilon^*_2 = (0\ 0\ 0,\ 0\ 0\ -).$$

Applying procedure B to check the existence of MOCO alterms, one has

1. The alterm $A_1 = (x_1 \vee x_2 \vee x_3)$ with $\epsilon'_2 = (- - 0)$ as the only one in S_1. Thus I = 1, and $\ell(1) = 3$.
2. Set r = 1 and i = 1.
3. Obtain $T_1 = \bar{A}_1 = \bar{x}_1\ \bar{x}_2\ \bar{x}_3$.
4. $\bar{x}_1\ \bar{x}_2\ \bar{x}_3$ subsumes a term $\bar{x}_1\ \bar{x}_2$ in the basic set of MO elements of Table 3.3.
5. Thus $(x_1 \vee x_2 \vee x_3)$ is a MOCO alterm whose $\epsilon^*_2 = (- - 0,\ 0\ 0\ -)$.
6. Literal $\bar{x}_3$ is removed from T_1, $\bar{x}_1\ \bar{x}_2$ is obtained. Set r = 2. By removing $\bar{x}_1$ and $\bar{x}_2$, we get new T_1's i.e., $\bar{x}_2\ \bar{x}_3$ and $\bar{x}_1\ \bar{x}_3$. But these will be rejected in step 4.
4. (second time) $\bar{x}_1\ \bar{x}_2$ is a MOPI term in the basic set of MO elements of

Table 3.3.

5. (second time) Thus $x_1 \vee x_2$ is a MOCO alterm whose $\overset{*}{\epsilon}_2 = (- - 0,\ 0\ 0\ -)$.

6. (second time) Since $\ell(1)\text{-}r = 1 \not> 1$ and $i + 1 = 2 \not\leq I$, the procedure terminates.

Two MOCO alterms are found. They are:

$$(x_1 \vee x_2 \vee x_3) \qquad \epsilon_2^* = (- - 0,\ 0\ 0\ -)$$

$$(x_1 \vee x_2) \qquad \epsilon_2^* = (- - 0,\ 0\ 0\ -).$$

Applying procedure C to check the existence of MOCO terms, one has

1. The alterm $A_1 = x_3 \vee \bar{x}_4 \vee \bar{x}_5$ with $\epsilon''_2 = (0\ 0\ -)$ is the only one in S_2. Thus $I = 1$ and $\ell(1) = 3$.

2. Set $r = 1$ and $i = 1$.

3. Obtain $T_1 = \bar{A}_1 = \bar{x}_3\, x_4\, x_5$.

4. $\bar{x}_3\, x_4\, x_5$ subsumes a term $x_4\, x_5$ in the basic set of MO elements of Table 3.3.

5. Thus $\bar{x}_3\, x_4\, x_5$ is a MOCO term whose $\epsilon_2^* = (- 0\ 0,\ 0\ 0\ -)$.

6. Literal $\bar{x}_3$ is removed from T_1, and $x_4\, x_5$ is obtained.

Set $r = 2$. By removing x_4 and x_5 from T_1, we get new T_1's i.e., $\bar{x}_3\, x_4$ and $\bar{x}_3\, x_5$. But these will be rejected in step 4 (second time).

4. (second time) $x_4\, x_5$ is a MOPI term in the basic set of MO elements of Table 3.3.

5. (second time) Thus $x_4\, x_5$ is a MOCO term whose $\epsilon_2^* = (- 0\ 0,\ 0\ 0\ -)$.

6. (second time) Since $\ell(1)\text{-}r = 1 \not> 1$ and $i + 1 = 2 \not\leq I = 1$, the procedure terminates.

The MOCO terms are:

$$\bar{x}_3\, x_4\, x_5 \qquad \epsilon_2^* = (- 0\ 0,\ 0\ 0\ -)$$

$$x_4\, x_5 \qquad \epsilon_2^* = (- 0\ 0,\ 0\ 0\ -).$$

Therefore, the sufficient set of MO elements is obtained, which contains:

MOPI terms: as shown in Table 3.3.

MOCO terms: $\bar{x}_3\, x_4\, x_5$, $\epsilon_2^* = (- \ 0 \ 0, \ 0 \ 0 \ -)$.

$x_4\, x_5$, $\epsilon_2^* = (- \ 0 \ 0, \ 0 \ 0 \ -)$.

MOCO alterms: $x_1 \vee x_2 \vee x_3$, $\epsilon_2^* = (- \ - \ 0, \ 0 \ 0 \ -)$.

$x_1 \vee x_2$, $\epsilon_2^* = (- \ - \ 0, \ 0 \ 0 \ -)$.

MOPI alterms $(x_1 \vee x_2 \vee x_3)$ and $(x_3 \vee x_4 \vee x_5)$ are not included in the sufficient set of MO elements because of Theorem 2.4.1.

3.4 The Selection of a Minimal Covering

The minimization of a single Boolean function can be converted into a covering problem after the set of prime implicants is found. McCluskey [32] and Cobham et al [10] formulated the covering problem by the integer linear programming method. The solution of the integer programming problem guarantees that the solution of the covering problem has the minimum cost under some specific cost function.

The application of the integer linear programming method to the minimization of multiple-output functions is an extension of that to the minimization problem of a single output function mentioned above. However, the following should be incorporated into the formulation of an integer linear programming problem.

1. Since an output function can either be in OR-JOINT or be in AND-JOINT, all possible grouping patterns of all functions into OR-JOINT and AND-JOINT must be taken into consideration.
2. The consideration of all MOPI terms, MOPI alterms, MOCO terms, and MOCO alterms.
3. The cost consideration of each MOPI term, MOPI alterm, MOCO term, and MOCO alterm.
4. The cost consideration of the objective function of the integer linear programming problem in terms of the cost function of the network to be designed.

In the following, the idea of minimization of a single Boolean function formulated in the form of an integer linear programming problem is first presented. Then, the extension to the minimization of multiple-output functions is given.

It will be seen later that the integer linear programming problems for the minimization of multiple-output functions are a special class of all integer linear programming problems, because the solution vector of our integer linear programming problem for the minimization of multiple-output functions is a binary vector of 1 and 0. Furthermore, the coefficients of our integer linear programming problem are all integers. Those features should be taken into consideration when one is attempting to devise an efficient computational algorithm to solve this type of integer linear programming problem.

3.4.1 Integer Linear Programming Formulation for the Minimization of a Single Output Network. (10, 23)

An example in the following is used to illustrate the integer linear programming formulation for the minimization of a single output network.

Let A, B, and C be three prime implicants of a function f which consists of three fundamental products denoted by #1, #2 and #3. The implication relation between a prime implicant and a fundamental product is shown in the next table, where an entry 1 is placed whenever the fundamental product in a row subsumes the prime implicant in a column.

	Prime Implicants		
Fundamental Products	A	B	C
#1	1	1	0
#2	0	1	1
#3	1	0	1
Variables Assignments	y_1	y_2	y_3

Also, three variables y_1, y_2, and y_3 are assigned to prime implicants A, B and C, respectively.

If one simply wants to find a minimum number of prime implicants such that all fundamental products of the function f are covered, then this is actually a covering problem. The covering problem can be formulated as the following integer linear programming problem:

Objective function $z = y_1 + y_2 + y_3$ is to be minimized,

under constraints $y_1 + y_2 \geq 1$

$$y_2 + y_3 \geq 1$$

$$y_1 + y_3 \geq 1$$

$$y_1, y_2, y_3 = 1 \text{ or } 0.$$

The general form of an integer programming problem can be written as,

To minimize $z = \vec{c} \cdot \vec{y}$

under

$$A\vec{y} \geq \vec{b}$$

$\vec{y}$ = binary vector of 1 and 0.

Corresponding to the above example,

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} \qquad \vec{b} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \qquad \vec{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \qquad \vec{c}^T = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

The matrix A is called a covering coefficient matrix. For simplicity, ILP will mean "integer linear programming".

3.4.2 The ILP Formulation for the Minimization of Multiple-Output Networks Under a Specified Grouping Pattern.

As has been mentioned before, in the process of finding a minimal network realizing a set of output functions f_1, f_2, ..., f_m, all possible groupings of f_1, f_2, ..., f_m into OR-JOINT and AND-JOINT Boolean forms should be taken into consideration. Which of OR-JOINT and AND-JOINT

Boolean forms each of $f_1, f_2, \cdots, f_m$ is put into is called a grouping pattern. A grouping pattern of $f_1, f_2, \cdots, f_m$ is denoted by a vector $\vec{\sigma} = (\sigma_1, \sigma_2, \cdots, \sigma_m)$, where σ_i is either 1 or 0. $\sigma_i = 1$ if a function f_i is in the group of OR-JOINT Boolean form, $\sigma_i = 0$ if the function f_i is in the group of AND-JOINT Boolean form. There are 2^m possible grouping patterns of $f_1, f_2, \cdots, f_m$.

Assume that a minimal two-level AND-OR network realizing $f_1, f_2, \cdots, f_m$ is designed. Let this network be denoted by $\eta(\Phi_1, \Phi_2, \cdots, \Phi_m)$. Then let us call a grouping pattern corresponding to the network $\eta(\Phi_1, \Phi_2, \cdots, \Phi_m)$ an optimal grouping pattern of $f_1, f_2, \cdots, f_m$, denoted by $\vec{\sigma}^0 = (\sigma^0_1, \sigma^0_2, \cdots, \sigma^0_m)$, where $\sigma^0_i = 1$ if $f_i = \Phi_i$ is an OR-JOINT and $\sigma^0_i = 0$ if $f_i = \Phi_i$ is an AND-JOINT. Although it is desirable to know a priori the optimal grouping pattern of the given set of output functions before starting to design a minimal network, the optimal grouping pattern will not be known until the minimal network is determined.

In the conventional McCluskey minimization method, the optimal grouping pattern is always assumed to be $(1, 1, \cdots, 1)$. The minimal multiple-output two-level network is then to be designed based on this presumed optimal grouping pattern. However, as already mentioned, the assumption of optimal grouping pattern $(1, 1, \cdots, 1)$ does not necessarily hold true in all cases. Two methods of finding the optimal grouping pattern will be given in Section 3.4.4. In this section, the ILP method and its formulation for the minimization of a multiple-output network under a presumed grouping pattern $(\sigma_1, \sigma_2, \cdots, \sigma_m)$ for σ_i = either 1 or 0, $i = 1, 2, \cdots, m$ is to be described. This is a preliminary discussion which leads to the description of finding a minimal network by the ILP method given in Section 3.4.4. The network found by the ILP formulation in this section is not necessarily a truly minimal network in terms of the cost

function previously defined. Instead, it is a minimal network under the presumed grouping pattern. A truly minimal network can be found by the methods given in Section 3.4.4.

In the following discussion, we assume that a minimal network is to be found under a presumed grouping pattern of the set of output functions. Let this presumed grouping pattern be denoted by $(\sigma_1, \sigma_2, \ldots, \sigma_m)$ where σ_i = either 1 or 0 for i = 1, 2, ..., m. The presumed grouping pattern presets the Boolean form Φ_i of an output function f_i to an OR-JOINT if $\sigma_i = 1$ and to an AND-JOINT if $\sigma_i = 0$. However, it is easier to formulate an inequality of our ILP problem in terms of OR-JOINT instead of AND-JOINT (i.e., in the case of $\sigma_i = 0$ for some i). Hence, if $\sigma_i = 0$, $\bar{f}_i$ will be taken into consideration, and if $\sigma_i = 1$, f_i will be taken into consideration because the solution of the ILP problem gives only an OR-JOINT for either f_i or $\bar{f}_i$. If this OR-JOINT is for $\bar{f}_i$ (i.e., $\sigma_i = 0$), then the output $f_i = \Phi_i$ is an AND-JOINT. Otherwise, the output $f_i = \Phi_i$ remains as an OR-JOINT. The notation f_i^* will thus be used in this section as a preset form of an output function; $f_i^* = f_i$ if $\sigma_i = 1$, and $f_i^* = \bar{f}_i$ if $\sigma_i = 0$. Only the MO elements among the sufficient set of MO elements which imply f_i if $\sigma_i = 1$ or $\bar{f}_i$ if $\sigma_i = 0$ for each i = 1, 2, ..., m will be used in the discussion of ILP formulation. This is in contrast to that in Section 3.4.4 where all MO elements in the sufficient set of MO elements which imply a product of functions (or complemented functions) including f_i (or $\bar{f}_i$) are used when a truly minimal network is desired.

In the following, the formulation of the objective function will first be discussed. Then, the formulation of the set of inequalities which represents the main body of the ILP problem for the minimization of a multiple-output network is to be presented.

The main difficulty of formulating the objective function of the

ILP problem for the minimization of a multiple-output network lies in the determination of the cost of a MO element which can be shared by more than one output function. A MO element may be a MOPI term, a MOPI alterm, a MOCO term, or a MOCO alterm. The cost of such a MO element will be defined.

There is a MO element which may possibly be shared by r different output functions. Let those r output functions be $f_{i_1}, f_{i_2}, \ldots, f_{i_r}$. Introduce the variables $y_{i_k j_k}$, where $k \in \{1, 2, \ldots, n\}$, such that $y_{i_k j_k}$ is 1 when this MO element is connected to f_{i_k} and otherwise 0. If the number of literals of this MO element is ℓ, then the cost $\ell + 1$ when $\ell \geq 2$ and 0 when $\ell = 1$ is assigned to this MO element, since ℓ gate-inputs and a gate (AND or OR) are needed to realize this MO element if $\ell \geq 2$, and nothing is needed when $\ell = 1$.

The cost of this MO element in a network which realizes functions $f_{i_1}, f_{i_2}, \ldots, f_{i_r}$ and others is then expressed as follows.

To add $y_{i_1 j_1} + y_{i_2 j_2} + \ldots + y_{i_r j_r} + (\ell + 1)$ if $\ell \geq 2$,

or $y_{i_1 j_1} + y_{i_2 j_2} + \ldots + y_{i_r j_r}$ if $\ell = 1$

to the total (network) cost function, if

$$y_{i_1 j_1} + y_{i_2 j_2} + \ldots + y_{i_r j_r} \geq 1.$$

(i.e., if the MO element is connected to any of output functions.) If the latter inequality is not satisfied, i.e., $y_{i_1 j_1} = y_{i_2 j_2} = \ldots = 0$, then no cost is added.

Then introduce a constant δ which is determined for each MO element;

$$\delta = 1 \text{ if } \ell \geq 2 \text{ and } \delta = 0 \text{ if } \ell = 1.$$

The above additional cost can be stated in a single expression,

$$y_{i_1 j_1} + y_{i_2 j_2} + \ldots + y_{i_r j_r} + (\ell + 1)\,[1 - (1-y_{i_1 j_1})(1-y_{i_2 j_2}) \ldots (1-y_{i_r j_r})]\delta.$$

However, it is not linear. It can be expressed as a linear form by introducing a variable t, which is called a linearization variable.

$$t = [1-(1-y_{i_1 j_1})(1-y_{i_2 j_2}) \ldots (1-y_{i_r j_r})],$$

which is expressed by the following two inequalities;

$$y_{i_1 j_1} + y_{i_2 j_2} + \cdots + y_{i_r j_r} - t \geq 0$$
$$rt - (y_{i_1 j_1} + y_{i_2 j_2} + \cdots + y_{i_r j_r}) \geq 0$$

where t = either 1 or 0.

Then the above cost of the MO element is expressed as,

$$y_{i_1 j_1} + y_{i_2 j_2} + \cdots + y_{i_r j_r} + (\ell + 1) \cdot t \cdot \delta.$$

Hence, for every such MO element, the cost

$$y_{i_1 j_1} + y_{i_2 j_2} + \cdots + y_{i_r j_r} + (\ell + 1) \cdot t \cdot \delta$$

is added to the objective function (or cost function) of our ILP formulation. And append two inequalities

$$y_{i_1 j_1} + y_{i_2 j_2} + \cdots + y_{i_r j_r} - t \geq 0$$
$$rt - (y_{i_1 j_1} + y_{i_2 j_2} + \cdots + y_{i_r j_r}) \geq 0$$

to the set of inequalities of our ILP formulation.

For those MO elements which are shared by only one output function, its cost is simply $(\ell + 1)\, y_{i_k j_k}$, if $\ell \geq 2$, and $y_{i_k j_k}$, if $\ell = 1$. No inequality is needed in this case.

Thus, the objective function of our ILP formulation is to sum up all the terms

$$y_{i_1 j_1} + y_{i_2 j_2} + \cdots + y_{i_r j_r} + (\ell + 1)\, t \cdot \delta$$

for MO elements which are shared by more than one output function, and

$$y_{i_k j_k} \quad (\ell + 1) \quad \text{or} \quad y_{i_k j_k}$$

for MO elements which are shared by only one output function in the set $(f_1^*, f_2^*, \ldots, f_m^*)$.

Two additional inequalities are generated for each MO element which are shared by more than one output function. For simplicity sake, a matrix notation

$$B\, \vec{y}_0 \geq 0$$

will be used to represent the set of inequalities which are generated for all the MO elements which are shared by more than one output function, where the vector $\vec{y}_0$ corresponds to the vector consisting of all y's and t's variables assigned to those MO elements, and matrix $\mathbb{B}$ is a coefficient matrix. An example in Table 3.6 (b) shows the vector $\vec{y}_0$ and the numerical values of elements in $\mathbb{B}$. The set of inequalities $\mathbb{B}\,\vec{y}_0 \geq \vec{0}$ shall be appended to the main body of the ILP formulation which is to be described next. Three notations are given in the following in order to facilitate the formulation of the set of inequalities of our ILP problem.

Notation: The fundamental products will be denoted by FP_0, FP_1, ..., FP_{2^n-1}, where the subscript j of FP_j represents the decimal equivalence of the binary representation of the fundamental product. For example, $x_1 \bar{x}_2 \bar{x}_3 x_4$ is represented by 1 0 0 1 and denoted by FP_9.

Notation: A MO element is denoted by PI_{ij}, where the subscript i indicates that PI_{ij} implies a product of functions including f_i^* (i.e., either including f_i or $\bar{f}_i$), and j is a running index. The variable assigned to PI_{ij} is y_{ij}.

Notation: The cost of PI_{ij} is denoted by C_{ij}, where

$C_{ij} = 0$ if the number of literals of PI_{ij} is 1.

$= 1 + \ell$ if the number of literals ℓ of PI_{ij} is ≥ 2.

The covering coefficient matrix for an individual function should be defined first. Then the combination of all such covering coefficient matrices for all functions plus the set

$$\mathbb{B}\,\vec{y}_0 \geq \vec{0}$$

will represent the set of inequalities of the ILP formulation.

Notation: The covering coefficient matrix for a function f_i^* is $\mathbb{A}_i = \{a_{jk}^{(i)}\}$, where

$a_{jk}^{(i)} = 1$ if FP_k of f_i^* implies PI_{ij}

$= 0$ otherwise.

For a function f_i^*, the set of inequalities which characterizes the covering problem is represented by

$$\mathbb{A}_i \vec{y}_i \geq \vec{1},$$

where $\mathbb{A}_i = \{a_{jk}^{(i)}\}$ and $\vec{y}_i^T = (y_{i1}, y_{i2}, \ldots, y_{is}, \ldots)$ and y_{ij}'s are the variables assigned to the PI_{ij}'s of f_i^*, and superscript T is to denote the transpose of a vector.

The covering coefficient matrix $\mathbb{A}_i$ is obtained in the same way as that in Section 3.4.1.

Based on each $\mathbb{A}_i$ formed separately, a matrix A;

$$\mathbb{A} = \begin{bmatrix} \mathbb{A}_1 & & & \\ & \mathbb{A}_2 & & \\ & & \mathbb{A}_3 & \\ & & & \ddots \\ & & & \mathbb{A}_m \end{bmatrix}$$

is introduced. There are m such $\mathbb{A}_i$'s, one for each function in the set of $\{f_1^*, f_2^*, \ldots, f_m^*\}$.

Similarly,

$$\vec{y}_0^T = (\vec{y}_1^T, \vec{y}_2^T, \ldots, \vec{y}_m^T, \vec{t}^T),$$

where $\vec{y}_i$, i = 1, 2, ..., m and $\vec{t} = (t_1, t_2, \ldots, t_s, \ldots)$ were already defined.

The set of inequalities,

$$\mathbb{A} \vec{y}_0 \geq \vec{1}$$

combines with the set of inequalities

$$\mathbb{B} \vec{y}_0 \geq \vec{0}$$

generated from those MO elements which can be shared by more than one output function as described before. Then,

$$\mathbb{A}_0 \vec{y}_0 \geq \vec{b}_0$$

is formed where

$$\mathbb{A}_0 = \left[\begin{array}{c|c} \mathbb{A} & 0 \\ \hline \mathbb{B} & \end{array}\right], \qquad \vec{b}_0 = \begin{bmatrix} \vec{1} \\ \vec{0} \end{bmatrix}.$$

The matrix inequality represents the complete set of inequalities of our minimization problem.

In summary, the minimal covering problem (i.e., the minimization problem) for multiple-output functions can be formulated in the following integer linear programming problem:

Minimize the objective function

$$z = m + \sum_{i,\,j} C_{ij}\, y_{ij} + \sum_{r} (1 + \ell_r\, \delta_r)\, t_r$$

under the constraints

$$\mathbb{A}_0\, \vec{y}_0 \geq \vec{b}_0,$$

where

$$\mathbb{A}_0 = \left[\begin{array}{c|c} \mathbb{A} & 0 \\ \hline \mathbb{B} & \end{array}\right], \qquad \vec{y}_0 = \begin{bmatrix} \vec{y} \\ \vec{t} \end{bmatrix}, \qquad \vec{b}_0 = \begin{bmatrix} \vec{1} \\ \vec{0} \end{bmatrix}.$$

and $\vec{y}_0$ is the vector of (0, 1) - variables, C_{ij} is the cost of y_{ij} and, $\delta_r = 1$ if $\ell_r \geq 2$, $\delta_r = 0$ if $\ell_r = 1$. t_r is a linearization variable.

One will see in the following example that the majority of coefficients of $\mathbb{A}$ and $\mathbb{B}$ are 1 and 0 except a minor portion of $\mathbb{B}$ which may be integers other than 1 or 0. The matrix $\mathbb{A}_0$ shown above is always decomposable into m + 1 submatrices.

In order to show the procedure of the ILP formulation, an example is given in the following. This is the minimization problem of three functions f_1, f_2 and f_3 under the preset grouping pattern (1 1 0). Thus f_1, f_2, and $\bar{f}_3$ are used.

Assume that all MO elements which imply f_1, f_2 and $\bar{f}_3$ are given as follows.

For f_1, the PI's are PI_{11}, PI_{12}, PI_{13}; for f_2, PI_{21}, PI_{22}, and PI_{23}; and for $\bar{f}_3$, PI_{31} and PI_{32}. The covering table of this problem is shown in Table 3.5. The variable assignments are at the bottom row.

The complete ILP formulation in $A_0 \vec{y}_0 \geq \vec{b}_0$ form is shown in Table 3.6 (a); the decomposed form is shown in Table 3.6 (b).

Table 3.5. The Covering Table

	f_1			f_2			$\bar{f}_3$	
	PI_{11}	PI_{12}	PI_{13}	PI_{21}	PI_{22}	PI_{23}	PI_{31}	PI_{32}
FP_0		1					1	1
FP_1		1					1	1
FP_2				1			1	
FP_3				1			1	
FP_4	1				1			1
FP_5							1	1
FP_6	1			1	1			
FP_7				1			1	
FP_8							1	
FP_9							1	
FP_{10}				1			1	
FP_{11}				1			1	
FP_{12}	1		1		1			
FP_{13}							1	
FP_{14}	1		1	1	1	1		
FP_{15}				1			1	
Variables Assignments	y_{11}	y_{12}	y_{13}	y_{21}	y_{22}	y_{23}	y_{31}	y_{32}

The following is assumed:

$PI_{11} = PI_{22}$, $PI_{13} = PI_{23}$ are MOPI terms of 2 and 3 literals, respectively.

$PI_{11} = PI_{22} = \overline{PI}_{31}$ are MOCO terms of 2 literals. The linearization variables assigned to $PI_{11} = \overline{PI}_{31}$ is t_1 and to PI_{13} is t_2. PI_{12}, PI_{21}, PI_{32} have 3, 1, 2 literals, respectively.

Table 3.6 (a). ILP Formul.tion for the Problem in Table 3.5.

Minimization of

$$z = 3 + y_{11} + 4y_{12} + y_{21} + y_{22} + y_{23} + y_{31} + 3y_{32} + 3t_1 + 4t_2$$

under

$$A_0 \vec{y}_0 \geq \vec{b}_0$$

$$\begin{pmatrix}
0 & 1 & 0 & & & & & & & & \\
0 & 1 & 0 & & & & & & & & \\
1 & 0 & 0 & & & & & & & & \\
1 & 0 & 0 & & & & & & & & \\
1 & 0 & 1 & & & & & & & & \\
1 & 0 & 1 & & & & & & & & \\
\hline
 & & & 1 & 0 & 0 & & & & & \\
 & & & 1 & 0 & 0 & & & & & \\
 & & & 0 & 1 & 0 & & & & & \\
 & & & 1 & 1 & 0 & & & & & \\
 & & & 1 & 0 & 0 & & & & & \\
 & & & 1 & 0 & 0 & & & & & \\
 & & & 1 & 0 & 0 & & & & & \\
 & & & 0 & 1 & 1 & & & & & \\
 & & & 1 & 1 & 1 & & & & & \\
 & & & 1 & 0 & 0 & & & & & \\
\hline
 & & & & & & 1 & 1 & & & \\
 & & & & & & 1 & 1 & & & \\
 & & & & & & 1 & 0 & & & \\
 & & & & & & 1 & 0 & & & \\
 & & & & & & 0 & 1 & & & \\
 & & & & & & 1 & 1 & & & \\
 & & & & & & 1 & 0 & & & \\
 & & & & & & 1 & 0 & & & \\
 & & & & & & 1 & 0 & & & \\
 & & & & & & 1 & 0 & & & \\
 & & & & & & 1 & 0 & & & \\
 & & & & & & 1 & 0 & & & \\
\hline
1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & -1 & 0 & \\
-1 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 3 & 0 & \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & \\
0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 2 &
\end{pmatrix}
\begin{pmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{31} \\ y_{32} \\ t_1 \\ t_2 \end{pmatrix}
\geq
\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Table 3.6 (b). The Decomposed Form of ILP Formulation

The matrix $\mathbb{A}_0$ and vector $\vec{y}$ in Table 3.6 (a) are decomposed into the following forms:

$$\mathbb{A}_0 = \left[\begin{array}{ccc|c} \mathbb{A}_1 & & & 0 \\ & \mathbb{A}_2 & & \\ & & \mathbb{A}_3 & \\ \hline & \mathbb{B} & & \end{array}\right] \qquad \vec{y}_0 = \begin{bmatrix} \vec{y}_1 \\ \vec{y}_2 \\ \vec{y}_3 \\ \vec{t} \end{bmatrix}$$

Then $\mathbb{A}_0 \vec{y} \geq 1$ is decomposed into the following four submatrices:

$$\mathbb{A}_1 \vec{y}_1 \geq \vec{1}$$

$$\begin{bmatrix} 0&1&0 \\ 0&1&0 \\ 1&0&0 \\ 1&0&0 \\ 1&0&1 \\ 1&0&1 \end{bmatrix} \begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

$$\mathbb{A}_2 \vec{y}_2 \geq \vec{1}$$

$$\begin{bmatrix} 1&0&0 \\ 1&0&0 \\ 0&1&0 \\ 1&1&0 \\ 1&0&0 \\ 1&0&0 \\ 1&0&0 \\ 0&1&0 \\ 1&1&1 \\ 1&0&0 \end{bmatrix} \begin{bmatrix} y_{21} \\ y_{22} \\ y_{23} \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

$$\mathbb{A}_3 \vec{y}_3 \geq \vec{1}$$

$$\begin{bmatrix} 1&1 \\ 1&1 \\ 1&0 \\ 1&0 \\ 0&1 \\ 1&1 \\ 1&0 \\ 1&0 \\ 1&0 \\ 1&0 \\ 1&0 \\ 1&0 \\ 1&0 \end{bmatrix} \begin{bmatrix} y_{31} \\ y_{32} \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

$$\mathbb{B} \vec{y}_0 \geq \vec{0}$$

$$\begin{bmatrix} 1&0&0&0&1&0&1&0&-1&0 \\ -1&0&0&0&-1&0&-1&0&3&0 \\ 0&0&1&0&0&1&0&0&0&-1 \\ 0&0&-1&0&0&-1&0&0&0&2 \end{bmatrix} \begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{31} \\ y_{32} \\ t_1 \\ t_2 \end{bmatrix} \geq \vec{0}$$

3.4.3 Techniques to Reduce th Size of the ILP Formulation

From the example just shown in Table 3.6, one discovers that the number of inequalities (i.e., the size of matrix A_0) is large and that the number of redundant inequalities (e.g., the 2nd, 4th and 6th inequalities in Table 3.6 (a)) is also large. Both of these facts directly affect the computability for solving this type of ILP problems. Thus let us derive some reduction techniques to delete the redundant inequalities and accordingly to reduce the size of the matrix A_0.

Three reduction techniques are to be given here. Each of them is described by a theorem. Some notations are needed before the theorems can be stated.

Notation: The collection of PI_{ij} for a particular i which are implied by the same fundamental product FP_k is denoted by $C_i\ (PI_{ij} \leftarrow FP_k)$.

Notation: The collection of all FP_j's of a particular f_i^* which imply the same PI_{ik} is to be denoted by $C_i\ (FP_j \rightarrow PI_{ik})$.

It should be noted that in $C_i\ (PI_{ij} \leftarrow FP_k)$ the indices i and k are fixed. For example, $C_i\ (PI_{ij} \leftarrow FP_k)$ represents the set $\{PI_{ij_1}, PI_{ij_2}, \ldots, PI_{ij_r}\}$, where each PI_{ij_s} is implied by FP_k for $s \in \{1, 2, \ldots, r\}$.

Three reduction techniques stated in the following three theorems will be used extensively and repeatedly in reducing the size of the ILP formulation.

Theorem 3.4.1: Let $C_i\ (PI_{ij} \leftarrow FP_k)$ and $C_i\ (PI_{ij} \leftarrow FP_\ell)$ be two sets of PI_{ij}'s. If $C_i\ (PI_{ij} \leftarrow FP_k)$ is a subset of $C_i\ (PI_{ij} \leftarrow FP_\ell)$, then the PI_{ij} of this $C_i\ (PI_{ij} \leftarrow FP_\ell)$ implied by the fundamental product FP_ℓ can be eliminated. (i.e., PI_{ij}'s located in column $C_i\ (PI_{ij} \leftarrow FP_\ell)$ and row FP_ℓ in Table 3.7.)

Proof: Since each PI_{ij} in the set $C_i\ (PI_{ij} \leftarrow FP_k)$ which is implied by

FP_k is also implied by FP_ℓ if $C_i\ (PI_{ij} \leftarrow FP_k)$ is a subset of $C_i\ (PI_{ij} \leftarrow FP_\ell)$, the set $C_i\ (PI_{ij} \leftarrow FP_k)$ is sufficient to cover the fundamental product FP_k as well as FP_ℓ.

Q.E.D.

Hence, the above theorem is to reduce the number of fundamental products and accordingly to reduce the number of rows of the matrix A_0.

The next theorem is to reduce the number of MO elements i.e., to reduce the number of columns of matrix A_0.

Theorem 3.4.2: Let $C_i\ (FP_j \rightarrow PI_{ik})$ and $C_i\ (FP_j \rightarrow PI_{i\ell})$ be two sets of fundamental products, and c_{ik}, $c_{i\ell}$ be the corresponding costs of PI_{ik} and $PI_{i\ell}$, respectively. If $C_i\ (FP_j \rightarrow PI_{i\ell})$ is a subset of $C_i\ (FP_j \rightarrow PI_{ik})$ and $c_{i\ell} > c_{ik}$, then the entries in $C_i\ (FP_j \rightarrow PI_{i\ell})$ for this ℓ can be eliminated.

Proof: This is because that PI_{ik} also implies all fundamental products in $C_j\ (FP_j \rightarrow PI_{i\ell})$ if $C_i\ (FP_j \rightarrow PI_{i\ell})$ is a subset of $C_i\ (FP_j \rightarrow PI_{ik})$. If $c_{i\ell} > c_{ik}$, the MO element $PI_{i\ell}$ can naturally be eliminated since there is no need to use more costly elements.

Q.E.D.

The next theorem is to remove the essential MO elements from the matrix A_0. But note that these MO elements must be included in a solution which will be obtained.

Theorem 3.4.3: If for a particular FP_k there exists exactly one element, say PI_{rs}, in the set $C_i\ (PI_{ij} \leftarrow FP_k)$ (i.e., PI_{rs} is an essential MO element), then this PI_{rs} should be included in a solution.

Proof: This PI_{rs} must be selected in a solution since otherwise no other PI_{ij} for some i implies the fundamental product FP_k.

Q.E.D.

The reduction techniques tated in the above three theorems can be repeatedly applied to a problem (e.g., Table 3.5) before the inequalities are formulated and until none is applicable. By repeated application of the above three reduction techniques all redundancies (of fundamental products and MO elements) are checked and eliminated.

The above reduction techniques are summarized in the following.

1. Assume that the set of PI_{ij} are given and that the fundamental products for each of the functions are known.
2. List the collections of $C_i\ (PI_{ij} \leftarrow FP_k)$ in the increasing order of k in a table according to the following format.

FP_k	$C_1\ (PI_{1j} \overset{f_1}{\leftarrow} FP_k)$	$C_2\ (PI_{2j} \overset{f_2}{\leftarrow} FP_k)$	----------	$C_m\ (PI_{mj} \overset{f_m}{\leftarrow} FP_k)$
FP_0				
FP_1				
FP_2				
FP_3				
.				
.				
.				
FP_{2^n-1}				

3. Do the elimination according to Theorem 3.4.1. If FP_k is eliminated, then the entries in row FP_k and column $C_i\ (PI_{ij} \leftarrow FP_k)$ are removed, where i is the index of f_i which contains FP_k.
4. Do the elimination according to Theorem 3.4.3. Record those PI_{ij}'s which are eliminated because they must be contained in a solution.
5. For each PI_{ik}, cost c_{ik} is assigned.
6. List the collections of $C_i\ (FP_j \rightarrow PI_{ik})$ in the increasing order of k in a table of the following format. No PI_{ij} eliminated in steps 3 and 4 are considered.

PI_{ik} k	C_1 $(FP_j \rightarrow PI_{1k})$ c_{1k}	C_2 $(FP_j \rightarrow PI_{2k})$ c_{2k}	----------	C_m $(FP_j \rightarrow PI_{mk})$ c_{mk}
1				
2				
3				
4				
.				
.				
.				

7. Do the elimination according to Theorem 3.4.2. If PI_{ik} is eliminated, then all the entries in row k and column C_i $(FP_j \rightarrow PI_{ik})$ are removed from the table.
8. Steps 2, 3, 4, 6, 7 are repeatedly applied until none of Theorems 3.4.1, 3.4.2, and 3.4.3 is applicable.
9. The remaining FP_k and PI_{ij} are used to form a new covering table such as shown in Table 3.5.
10. An ILP problem is to be formulated based on this covering table.

An example is shown in Table 3.7 to illustrate the use of reduction techniques to the example in Table 3.5.

Table 3.7.

(An example to illustrate the use of reduction techniques to the example shown in Table 3.5)

Step 1. FP_k's and PI_{ij}'s are given as in Table 3.5.

Step 2. A table is formed from Table 3.5 as follows.

FP_k	C_1 $(PI_{1j} \leftarrow FP_k)$	C_2 $(PI_{2j} \leftarrow FP_k)$	C_3 $(PI_{3j} \leftarrow FP_k)$
FP_0	PI_{12}		PI_{31}, PI_{32}
FP_1	PI_{12}		PI_{31}, PI_{32}
FP_2		PI_{21}	PI_{31}
FP_3		PI_{21}	PI_{31}
FP_4	PI_{11}	PI_{22}	PI_{32}
FP_5			PI_{31}, PI_{32}
FP_6	PI_{11}	PI_{21}, PI_{22}	
FP_7		PI_{21}	PI_{31}
FP_8			PI_{31}
FP_9			PI_{31}
FP_{10}		PI_{21}	PI_{31}
FP_{11}		PI_{21}	PI_{31}
FP_{12}	PI_{11}, PI_{13}	PI_{21}	
FP_{13}			PI_{31}
FP_{14}	PI_{11}, PI_{13}	PI_{21}, PI_{22}, PI_{23}	
FP_{15}		PI_{21}	PI_{31}

Step 3. By applying Theorem 3.4.1, the following PI_{ij}'s are removed from the above table.

In row FP_0: PI_{31}, PI_{32}. (because C_3 $(PI_{3j} \leftarrow FP_1)$ is a subset of $(C_3$ $(PI_{3j} \leftarrow FP_0)$.)

FP_1: PI_{12}, PI_{31}, PI_{32}.

FP_3: PI_{21}, PI_{31}.

FP_5 : PI_{31}, PI_{32}

FP_6 : PI_{11}, PI_{21}, PI_{22}.

FP_7 : PI_{21}, PI_{31}.

FP_8 : PI_{31}.

FP_9 : PI_{31}.

FP_{10}: PI_{21}, PI_{31}.

FP_{11}: PI_{21}, PI_{31}.

FP_{12}: PI_{11}, PI_{13}, PI_{22}.

FP_{13}: PI_{31}.

FP_{14}: PI_{11}, PI_{13}, PI_{21}, PI_{22}, PI_{23}.

FP_{15}: PI_{21}, PI_{31}.

Step 4. Essential MO elements:

PI_{11}, PI_{12} in column C_1 ($PI_{1j} \leftarrow FP_k$)

PI_{21}, PI_{22} in column C_2 ($PI_{2j} \leftarrow FP_k$)

PI_{31}, PI_{32} in column C_3 ($PI_{3j} \leftarrow FP_k$)

are removed to be included in a solution, Theorem 3.4.3.

Step 5. No more PI_{ij} and FP_k left. The algorithm thus terminates.

3.4.4 The ILP Method for Finding the Optimal Grouping Pattern

Two methods for finding the optimal grouping pattern are to be given in this section. The first method is a semi-exhaustive method which employs the branch-and-bound method developed for the solution of ILP problems to find the optimal grouping pattern. A small number of branchings is to be constructed. Each branching represents a presumed optimal grouping pattern, upon which the minimal network (This is the minimal for each branch, in other words locally minimal. But it may not be globally minimal over all branches.) is designed by the ILP method described in Sections 3.4.2 and 3.4.3. If the cost of a newly found network (i.e., the minimal network for each branch) is less than the lowest cost previously found, it is stored, replacing the latter

cost; otherwise, the latter is still stored. Each branch is tested by the same way until no branch is left. The number of branches should be kept as small as possible in order to make this method computationally feasible. Details are described in Appendix B.

The second method which is to be described later is straight-forward. The grouping pattern $\vec{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_m)$ is transformed into a set of binary variables which in turn are incorporated into the ILP formulation for finding a minimal covering as described in Sections 3.4.2 and 3.4.3. A solution of the ILP problem so formulated not only gives an optimal grouping pattern of the set of output functions to be realized but also gives the structure of the minimal network realizing the set of output functions.

The difference between the above two methods is as follows. The first method is to employ a series of small size ILP problems to find the optimal grouping pattern. The small size ILP problems may not be too time consuming to be solved by using the technique described in Appendix B. The second method is to employ a single relatively large size ILP problem to find the optimal grouping pattern and the structure of the minimal network. The solving of this single large size ILP problem may prove to be computationally infeasible if the number of (input) variables and the size of the sufficient set of MO elements are large. The comparison of the two methods has not been computationally tested.

In the following, let us describe the second method. The solution of the ILP by the second method not only gives the desired grouping pattern of the output functions, but also gives the structure of a minimal network. The ILP method is straight forward after the sufficient set of MO elements is found. The optimality of the network to be designed

by this ILP method is always guaranteed. The number of inequalities of this ILP program is m x 2^n. For large m, the number of given output functions, and for large n, the number of input variables, the solvability of the ILP program by computers poses a major drawback to this method.

Let FP_k be a fundamental product of f_i, and PI_{ik_1}, PI_{ik_2}, ..., PI_{ik_p} be members of the sufficient set of MO elements which are implied by this FP_k of f_i and which imply a product including f_i. Also, let FP_ℓ be a fundamental product of $\bar{f}_i$ and $PI_{i\ell_1}$, $PI_{i\ell_2}$, ..., $PI_{i\ell_q}$ be those MO elements in the sufficient set of MO elements which are implied by this FP_ℓ of $\bar{f}_i$ and imply a product of complemented functions including $\bar{f}_i$. Variables y_{ik_1}, y_{ik_2}, ..., y_{ik_p} are assigned to PI_{ik_1}, PI_{ik_2}, ..., PI_{ik_p} and variables $y_{i\ell_1}$, $y_{i\ell_2}$, ..., $y_{i\ell_q}$ are assigned to $PI_{i\ell_1}$, $PI_{i\ell_2}$, ..., $PI_{i\ell_q}$, respectively. Then the inequality corresponding to this FP_k is

(1) $y_{ik_1} + y_{ik_2} + \cdots + y_{ik_p} \geq 1$,

and the inequality corresponding to this FP_ℓ is

(2) $y_{i\ell_1} + y_{i\ell_2} + \cdots + y_{i\ell_q} \geq 1$.

Since each f_i in a minimal network η (Φ_1, Φ_2, ..., Φ_m) is <u>either</u> in OR-JOINT Boolean form <u>or</u> in AND-JOINT Boolean form, one needs only either inequality (1) or inequality (2) in the ILP formulation. If f_i is an OR-JOINT Boolean form, then only those fundamental products of f_i will be taken into consideration and the inequalities such as inequality (1) corresponding to those fundamental products will be used in the ILP formulation. If f_i is in AND-JOINT Boolean form, then only those fundamental products of $\bar{f}_i$ will be taken into consideration and the inequalities such as (2) corresponding to those fundamental products will be used in the ILP formulation.

The above "either or" statement can be simplified by introducing a new

variable Q_i, where

$Q_i = 1$ if f_i is in OR-JOINT Boolean form.

$= 0$ if f_i is in AND-JOINT Boolean form.

Thus, inequalities (1) and (2) are stated as:

If $Q_i = 1$, then $y_{ik_1} + y_{ik_2} + \cdots + y_{ik_p} \geq 1$ for $FP_k \in f_i$, and

if $Q_i = 0$, then $y_{i\ell_1} + y_{i\ell_2} + \cdots + y_{i\ell_q} \geq 1$ for $FP_\ell \in \bar{f}_i$.

If the variable Q_i is incorporated into the inequalities, the following two inequalities result:

(3) $y_{ik_1} + y_{ik_2} + \cdots + y_{ik_p} \geq 1 - U \cdot P_i$ for $FP_k \in f_i$

(4) $y_{i\ell_1} + y_{i\ell_2} + \cdots + y_{i\ell_q} \geq 1 - U \cdot Q_i$ for $FP_\ell \in \bar{f}_i$

where $P_i = 1 - Q_i$ and Q_i = either 1 or 0. U is a very large positive integer.

Further explanation of the use of the above two inequalities (3) and (4) is given in the following.

Suppose that f_i is found to be an OR-JOINT Boolean form in an optimal grouping pattern. Then $P_i = 0$ and $Q_i = 1$. Then inequality (3) becomes,

(5) $y_{ik_1} + y_{ik_2} + \cdots + y_{ik_p} \geq 1$

which is a normal covering inequality for a fundamental product of f_i.

And, the inequality (4) becomes

(6) $y_{i\ell_1} + y_{i\ell_2} + \cdots + y_{i\ell_q} \geq 1 - U$

which is always satisfied with

(7) $y_{i\ell_1} = y_{i\ell_2} = \cdots = y_{i\ell_q} = 0$,

because of the minimization of the objective function.

Thus, only the inequalities like (5) are actually retained in the ILP (program) since the other inequalities like (6) can be easily satisfied by a solution like (7).

If f_i is found to be in a Boolean AND-JOINT in an optimal grouping pattern; then the roles of inequalities (3) and (4) are reversed.

Both types of inequalities (3) and (4) are included in the ILP formulation. The ILP problem itself will determine whether $Q_i = 1$ or 0, which in turn determines whether the inequalities like (3) or (4) would be retained.

There are 2^n fundamental products for each function. Thus, each function has 2^n inequalities like (3) and (4). For m output functions, the total number of inequalities is m x 2^n in the matrix representation

$$A \vec{y}_0 \geq 1.$$

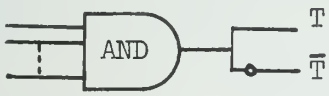
Solve the ILP problem for the minimization of a multiple-output network with the inequalities $A \vec{y}_0 \geq 1$ just formulated. The solution of the ILP problem not only gives the values of all y_{ij}'s but also gives the values of all Q_i's. The values of y_{ij}'s determine the structure of the minimal network which realizes the set of m output functions. The values of Q_i's determine the optimal grouping pattern of the set of m output functions.

It should be noted that the above ILP method is applied only after the reduction techniques described in Section 3.4.3 have been exhausted.

4. MINIMIZATION OF A NETWORK WHICH HAS INVERTERS

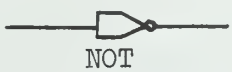
4.1 Introduction

The minimization procedure described in Chapter 3 assumes that no inverter is available. This chapter presents a minimization procedure when inverters are available. Two cases will be considered. The first case is the minimization with inverters which are available free of charge. Inverters are built-in at each of AND and OR gates which thus produces a complemented output as well as an uncomplemented output. Each of AND and OR gates is represented by



respectively, where $\bar{T}$ and T are the complemented output and the uncomplemented output of the AND gate, respectively, and $\bar{A}$ and A are the complemented and the uncomplemented outputs of the OR gate, respectively.

The second case is the minimization with NOT gates. The inversion is obtained through a NOT gate which is to be represented by



Each NOT gate has a cost of 2, since one gate and one gate-input are required in each NOT gate.

In order to use the algorithm developed in Chapter 3 with the least modification, the assumption will be made that a NOT gate does not constitute a "level" in a 2-level AND-OR network, i.e., NOT gates may be allowed between any first-level gate and any output gates.

In the following, an example will be shown to demonstrate the differences in the cost functions and in the network structures between the case when cost-free inversions are available and the case when the inversions are not available.

Three output functions are given as,

$f_1 (x_1, x_2, x_3, x_4) = \Sigma (0, 1, 4, 6, 12, 14)$

$f_2 (x_1, x_2, x_3, x_4) = \Sigma (2, 3, 5, 6, 7, 10, 11, 14, 15)$

$f_3 (x_1, x_2, x_3, x_4) = \Sigma (6, 7, 8, 9, 13, 14, 15).$

A minimal network realizing f_1, f_2 and f_3 is shown in Figure 4.1,

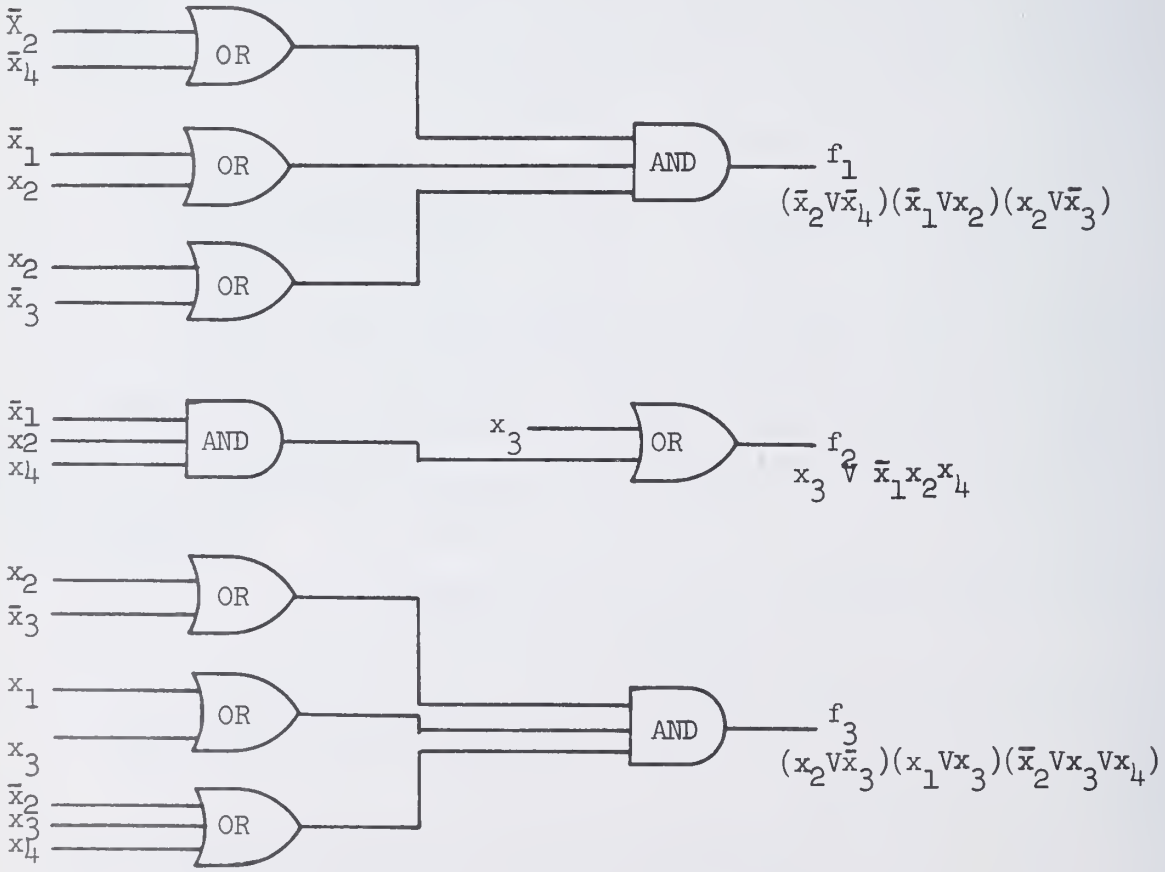


Figure 4.1

when no inversion is available. But when cost-free inversions are available, a minimal network realizing the same f_1, f_2, and f_3 is shown in Figure 4.2,

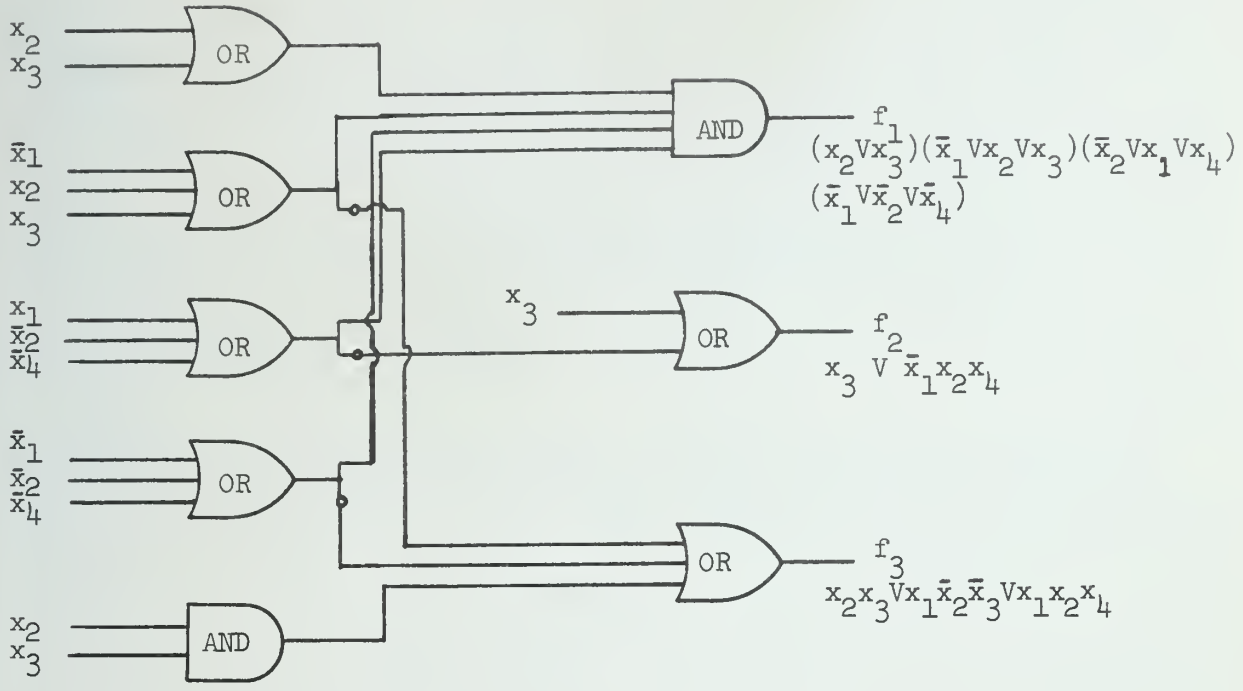


Figure 4.2

where some OR gates provide complemented outputs (with small circle o) and uncomplemented output (without small circle o) as well. In comparison of network cost, the first network has a cost 34 and the second one has a cost 30.

4.2 Minimization with Cost-Free Inversions

In this section, the assumption is made that each of AND and OR gates provides a complemented output as well as an uncomplemented output. Thus, whenever a term like $x^*_{i_1} x^*_{i_2} \ldots x^*_{i_k}$ is produced by an AND gate, an alterm $\bar{x}^*_{i_1} \vee \bar{x}^*_{i_2} \vee \ldots \vee \bar{x}^*_{i_k} = \overline{x^*_{i_1} x^*_{i_2} \ldots x^*_{i_k}}$ can be obtained through its complemented output. Similarly, whenever an alterm $x^*_{i_1} \vee x^*_{i_2} \vee \ldots \vee x^*_{i_k}$ is produced by an OR gate, a term $\bar{x}^*_{i_1} \bar{x}^*_{i_2} \ldots \bar{x}^*_{i_k} = \overline{x^*_{i_1} \vee x^*_{i_2} \vee \ldots \vee x^*_{i_k}}$ can also be obtained through its complemented output.

Those alterms and terms will be specially considered in addition to the MOPI terms, MOPI alterms, MOCO terms, and MOCO alterms as defined before.

Let us call a term $T = x_{i_1}^* x_{i_2}^* \ldots x_{i_k}^*$ and an alterm $A = \bar{x}_{i_1}^* \vee \bar{x}_{i_2}^* \vee \ldots \vee \bar{x}_{i_k}^* = \bar{T}$ a <u>T-A pair</u> if there exists an AND gate which produces T and A through its uncomplemented and complemented output, respectively. Similarly define an <u>A-T pair</u> if there exists an OR gate which produces A and T through its uncomplemented and complemented output, respectively. For example, in Figure 4.2 $A = \bar{x}_1 \vee x_2 \vee x_3$ and $T = x_1 \bar{x}_2 \bar{x}_3$ are an A-T pair. The number of literals k is restricted to $k \geq 2$.

In the minimization with cost-free inversion, there are two different kinds of T-A pairs and two different kinds of A-T pairs which are defined in the following.

<u>Definition 4.2.1</u>: A T-A pair is called a <u>MOPI T-A pair</u> for the set of output functions $f_1, f_2, \ldots, f_m$ if there exist two disjoint subsets of functions $\{f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}\}$ and $\{f_{p_1}, f_{p_2}, \ldots, f_{p_q}\}$ of $\{f_1, f_2, \ldots, f_m\}$ such that

(1) T is a prime implicant of the product of functions,

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} \text{ (or } \bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}\text{), and}$$

(2) A implies the product of functions,

$$f_{p_1} \cdot f_{p_2} \cdot \ldots \cdot f_{p_q} \text{ (or } \bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}\text{).}$$

<u>Definition 4.2.2</u>: An A-T pair is called a <u>MOPI A-T pair</u> for the set of output functions $f_1, f_2, \ldots, f_m$ if there exist two disjoint subsets of functions $\{f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}\}$ and $\{f_{p_1}, f_{p_2}, \ldots, f_{p_q}\}$ of $\{f_1, f_2, \ldots, f_m\}$ such that

(1) A is a MOPI alterm of the product of functions,

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} \text{ (or } \bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}\text{), and}$$

(2) T implies the product of functions,

$$f_{p_1} \cdot f_{p_2} \cdot \ldots \cdot f_{p_q} \text{ (or } \bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}\text{).}$$

Definition 4.2.3: A T-A pair is called a MOCO T-A pair for the given set of output functions $f_1, f_2, \ldots, f_m$ if there exist two disjoint subsets of functions $\{f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}\}$ and $\{f_{p_1}, f_{p_2}, \ldots, f_{p_q}\}$ such that

(1) T implies the product of functions,

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}, \text{ and}$$

(2) $\bar{A}$ implies the product of the complemented functions,

$$\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$$

where $\ell, q \geq 1$.

Definition 4.2.4: An A-T pair is called a MOCO A-T pair for the set of output functions $f_1, f_2, \ldots, f_m$ if there exist two disjoint subsets of functions $\{f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}\}$ and $\{f_{p_1}, f_{p_2}, \ldots, f_{p_q}\}$ such that

(1) A implies the product of functions

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$$

(2) $\bar{T}$ implies the product of the complemented functions

$$\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}.$$

where $\ell, q \geq 1$.

In Figure 4.3 a network $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_4)$ is shown which has outputs

$$f_1 = \Phi_1 = x_1 \bar{x}_2 x_3 \vee \bar{x}_3 x_4 x_5$$

$$f_2 = \Phi_2 = (\bar{x}_1 \vee x_2 \vee \bar{x}_3) \vee x_6 \vee x_4 x_5$$

$$f_3 = \Phi_3 = (x_1 \bar{x}_2 x_3)\ (x_3 \vee \bar{x}_4 \vee \bar{x}_5)$$

$$f_4 = \Phi_4 = x_1 x_2 \vee (\bar{x}_4 \vee \bar{x}_5).$$

We see that the AND gate which has inputs $\bar{x}_3$, x_4 and x_5 produces a MOCO T-A pair where $T = \bar{x}_3 x_4 x_5$ and $A = x_3 \vee \bar{x}_4 \vee \bar{x}_5$ for f_1 and f_3, respectively. The T-A pair, $T = \bar{x}_3 x_4 x_5$ and $A = x_3 \vee \bar{x}_4 \vee \bar{x}_5$, is a MOCO T-A pair because $T = \bar{x}_3 x_4 x_5$ produced from the uncomplemented output of the AND gate implies f_1 and the alterm $A = x_3 \vee \bar{x}_4 \vee \bar{x}_5$ produced from the complemented output of the AND gate is implied by f_3.

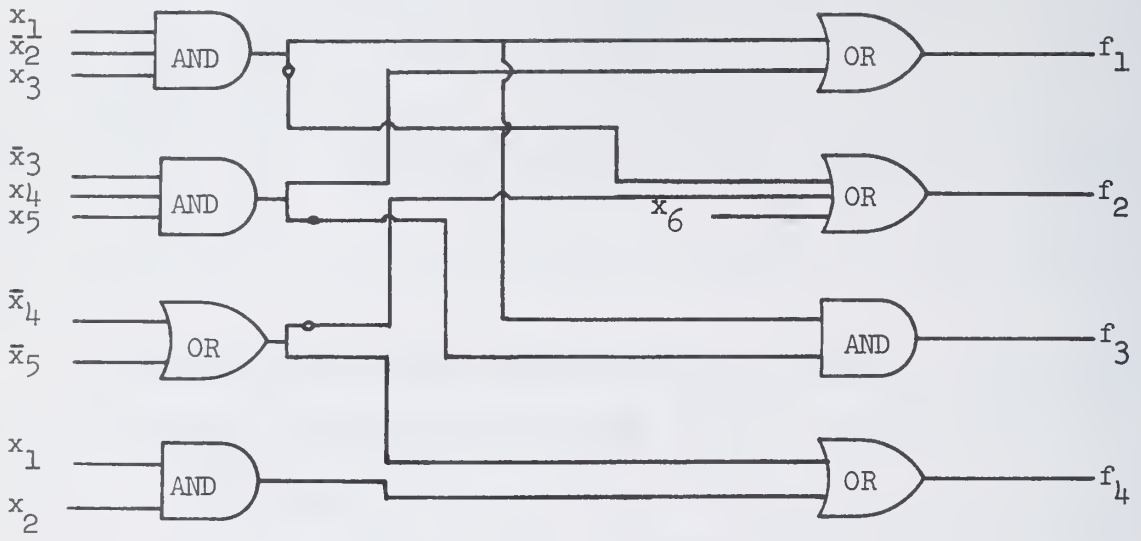


Figure 4.3 A Network η $(\Phi_1, \Phi_2, \Phi_3, \Phi_4)$

It should be noted that if the alterm A of a MOPI T-A pair and the term T of a MOPI A-T pair were never used for any output function in a network, the MOPI T-A pair and MOPI A-T pair are then simply degenerated to and considered as a MOPI term and a MOPI alterm, respectively.

Some properties concerning the existence of a MOPI T-A pair, a MOPI A-T pair, a MOCO T-A pair, and MOCO A-T pair are given in the following lemmas.

Lemma 4.2.1: Let $\{f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}\}$ and $\{f_{p_1}, f_{p_2}, \ldots, f_{p_q}\}$ be two disjoint subsets of output functions. Then the existence of the following implication relations,

either $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} \supseteq \bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$,

or $\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell} \supseteq f_{p_1} \cdot f_{p_2} \cdot \ldots \cdot f_{p_q}$

implies the existence of a MOPI T-A pair for the set of output functions $\{f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}, f_{p_1}, f_{p_2}, \ldots, f_{p_q}\}$.

Proof: If $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} \supseteq \bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}$, then there exists a term T such that

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} \supseteq T \supseteq \bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q}.$$

The first half of the above implication relation gives

$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} \supseteq T$, and the second half gives $\bar{f}_{p_1} \cdot \bar{f}_{p_2} \cdot \ldots \cdot \bar{f}_{p_q} \subseteq T$ which results in $f_{p_1} \cdot f_{p_2} \cdot \ldots \cdot f_{p_q} \supseteq \bar{T} = A$. By definition, this T-A pair is a MOPI T-A pair.

The proof for the other part is similar.

Q.E.D.

Lemma 4.2.2: Let $\{f_{j_1}, f_{j_2}, \ldots, f_{j_\ell}\}$ and $\{f_{p_1}, f_{p_2}, \ldots, f_{p_q}\}$ be two subsets of output functions. If there exists a MOCO T-A term with respect to these two subsets of functions, the $g_1 \cdot g_2 \subset g_1$ and $g_1 \cdot g_2 \subset g_2$, where $g_1 = f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ and $g_2 = f_{p_1} \cdot f_{p_2} \cdot \ldots \cdot f_{p_q}$.

Proof: By definition, this MOCO T-A pair and these two subsets of functions have the following relations:

$$T \subseteq f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell} = g_1$$
$$A \supseteq f_{p_1} \cdot f_{p_2} \cdot \ldots \cdot f_{p_q} = g_2$$

where T and A is the MOCO T-A pair, and $A = \bar{T}$. The results follow immediately by multiplying g_1 and g_2.

Q.E.D.

Lemma 4.2.3: Let $\{f_{i_1}, f_{i_2}, \ldots, f_{i_p}\}$, $\{f_{j_1}, f_{j_2}, \ldots, f_{j_q}\}$ and $\{f_{k_1}, f_{k_2}, \ldots, f_{k_r}\}$ be three disjoint subsets of output functions. If there exists a MOPI T-A pair with respect to the first two subsets of functions where A is also a MOCO alterm with respect to the last two subsets of functions, then this T-A pair is also a MOCO T-A pair with respect to the first and last subsets of functions.

Proof: By Definition 4.2.1, if a T-A pair is a MOPI T-A pair with respect to the first two subsets of functions, then the T-A pair has the following

properties:

(1) T is a prime implicant of the product of functions,

$$f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p}, \text{ and}$$

(2) $A = \overline{T}$ implies the product of functions,

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q}.$$

If this alterm $A = \overline{T}$ is also a MOCO alterm with respect to the last two subsets of functions, then

(1') A implies the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q}$, and

(2') $\overline{A} = T$ implies the product of the complemented functions

$$\overline{f}_{k_1} \cdot \overline{f}_{k_2} \cdot \ldots \cdot \overline{f}_{k_r}.$$

But a prime implicant always implies the function from which it is derived. Hence, by the condition (1) and (2') above, and by Definition 4.2.3 this T-A pair is a MOCO T-A pair with respect to two subsets of functions $\{f_{i_1}, f_{i_2}, \ldots, f_{i_p}\}$ and $\{f_{k_1}, f_{k_2}, \ldots, f_{k_r}\}$.

Q.E.D.

Actually, concerning the above three disjoint subsets of functions and the existence of the MOPI T-A pair, the MOCO T-A pair and the MOCO alterm, a stronger relation can be established as stated in the following lemma.

Lemma 4.2.4: Assume that there exist a MOPI T-A pair, a MOCO T-A pair, and a MOCO alterm, where the MOPI T-A pair and the MOCO T-A pair have the same T-A pair and the A of this T-A pair is the MOCO alterm. Then among these three, the MOPI T-A pair, the MOCO T-A pair, and the MOCO alterm, the existence of either two implies the existence of the other one.

Proof: If the MOPI T-A pair and the MOCO T-A pair have the same T-A pair and the A of this T-A pair is the MOCO alterm, then there exist three disjoint subsets of functions,

$$\{f_{i_1}, f_{i_2}, \ldots, f_{i_p}\}, \{f_{j_1}, f_{j_2}, \ldots, f_{j_q}\}, \text{ and } \{f_{k_1}, f_{k_2}, \ldots, f_{k_r}\}$$

such that the following three conditions hold:

(1) The T of this T-A pair is a prime implicant of $f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p}$.

(2) The $A = \bar{T}$ of this T-A pair implies $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q}$.

(3) The $\bar{A} = T$ of this T-A pair implies $\bar{f}_{k_1} \cdot \bar{f}_{k_2} \cdot \ldots \cdot \bar{f}_{k_r}$.

Lemma 4.2.3 proves that the existence of the MOPI T-A pair and the MOCO alterm implies the existence of the MOCO T-A pair. Other two cases will be proved in the following.

Conditions (1) and (2) gives that the T-A pair is a MOPI T-A pair. From condition (1) and (3) it follows that the T-A pair is a MOCO T-A pair. Hence, the existenceof the MOPI T-A pair and the MOCO T-A pair gives conditions (2) and (3) simultaneously, which implies the existence of a MOCO alterm (i.e., A).

Since the existence of the MOCO T-A pair and MOCO alterm A (which is identical to the A part of the MOCO T-A pair) only gives that

(1') T implies $f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p}$ (satisfied by the MOCO T-A pair.)

(2) $A = \bar{T}$ implies $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q}$ (satisfied by the MOCO alterm.)

we cannot immediately conclude that there exists a MOPI T-A pair. However, we can always find a prime implicant of $f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p}$ which is implied by T. Let this prime implicant be T'. Then the T'-A pair satisfies:

(1") T' is a prime implicant of $f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p}$, and

(2') $A' = \bar{T}'$ implies $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q}$.

By definition, this T'-A' pair is a MOPI T-A pair. Hence, the existence of the MOCO T-A pair and the MOCO alterm does imply the existence of a MOPI T-A pair.

Q.E.D.

Also, among the above three subsets of functions, two implication relations as shown in the following lemma can be established.

Lemma 4.2.5: If a T-A pair is a MOPI T-A pair and also is a MOCO T-A pair, then there exist three disjoint subsets of functions

$\{f_{i_1}, f_{i_2}, \ldots, f_{i_p}\}$, $\{f_{j_1}, f_{j_2}, \ldots, f_{j_q}\}$, and $\{f_{k_1}, f_{k_2}, \ldots, f_{k_r}\}$, such that

$$f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p} \supseteq \bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_q}$$
$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q} \supseteq f_{k_1} \cdot f_{k_2} \cdot \ldots \cdot f_{k_r}$$

hold.

Proof: If there exist such a MOPI T-A pair and a MOCO T-A pair, then three conditions (1), (2), and (3) in the proof of Lemma 4.2.5 hold. The condition (2) and (3) gives,

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q} \supseteq A \supseteq f_{k_1} \cdot f_{k_2} \cdot \ldots \cdot f_{k_r}.$$

The condition (2) also gives

$$\bar{A} = T \supseteq \bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_q},$$

which in combination with condition (1) results in

$$f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p} \supseteq T = \bar{A} \supseteq \bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_q}.$$

Q.E.D.

The above Lemma 4.2.5 could also have the following set of implication relations.

$$\bar{f}_{i_1} \cdot \bar{f}_{i_2} \cdot \ldots \cdot \bar{f}_{i_p} \supseteq f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_q}$$
$$f_{k_1} \cdot f_{k_2} \cdot \ldots \cdot f_{k_r} \supseteq f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_p},$$

if the MOPI T-A pair is obtained through the set of complemented functions as parenthesized in the conditions of Definition 4.2.1.

4.3 Algorithm

An algorithm for generating MOPI T-A pairs, MOPI A-T pairs, MOCO T-A pairs and MOCO A-T pairs will be given in this section. The basic set of MO elements for the set of output functions $f_1, f_2, \ldots, f_m$ is assumed to be found. From the basic set of MO elements, one can obtain two sets of alterms denoted by S_1 and S_2 in the algorithm for generating MOPI alterms

in Section 3.3. The definitions of S_1 and S_2 will be given here again. They will be used in the algorithms to be described next.

The set S_1 is the collection of all disjunctions of literals which are single literal terms in the basic set of MO elements. Every term in a disjunction of S_1 has the same ϵ'_2 part. The set S_2 is the same as S_1 except that the ϵ'_2 part is changed to its ϵ''_2 part.

1. Procedure for generating MOPI T-A pairs.

   (a) Denote the alterms in S_1 as $A_1, A_2, \ldots, A_I$. Each A_i has $\ell(i)$ literals for $i = 1, 2, \ldots, I$.

   Set $i = 1$.

   (b) For each A_i, obtain a $T_i = \overline{A}_i$.

   (c) Check whether or not T_i is a MOPI term in the basic set of MO elements whose ϵ'_2 has at least one dash. If yes, go to (f); otherwise go to (d).

   (d) If $\ell(i) \geq 3$, then set $\ell(i)$ to $\ell(i)$-1 and generate new T_i's by deleting one literal from T_i, one for each new T_i, and then go to (c). Otherwise, go to (e).

   (e) Set i to $i + 1$. If $i \leq I$, then go to (b). Otherwise, stop.

   (f) Record this T_i-A_i pair which is a MOPI T-A pair. The ϵ^*_2's for the T_i and A_i are also recorded.

   (g) Do steps (a) to (f) for the set S_2. Use ϵ''_2 instead of ϵ'_2.

2. Procedure for generating MOPI A-T pairs.

   The procedure is the same as the one for generating MOPI T-A pairs except step (e) is modified as

   (e') Check whether or not T_i subsumes/is a MOPI term whose ϵ'_2 has at least one dash in the basic set of MO elements. If yes, go to (f); otherwise, go to (e)

3. Procedure for generating MOSO A-T pairs.

(a) Denote the alterms in the set S_1 as

$$A_{11}, A_{12}, \cdots, A_{1I},$$

and the alterms in the set S_2 as

$$A_{21}, A_{22}, \cdots, A_{2p}.$$

(b) Detect if any alterm A_{1j}, $j \in \{1, 2, \ldots, I\}$ either subsumes or is subsumed by any alterm A_{2k}, $k \in \{1, 2, \ldots, p\}$.

If A_{1j} subsumes A_{2k}, then go to (c).

If A_{1j} is subsumed by A_{2k}, then go to (d).

Otherwise, stop.

(c) Record A_{2k} and its ϵ''_2 along with the ϵ'_1 of A_{1j}. Then go to (e).

(d) Record A_{1j} and its ϵ'_2 along with the ϵ''_2 of A_{2k}.

(e) Write down all alterms which are subsumed by either A_{2k} in (c) or A_{1j} in (d) along with its ϵ^*_2 of either A_{2k} or A_{1j}, respectively.

4. Procedure for generating MOPI T-A pairs.

(a) Denote the set of terms in the basic set of MO elements as

$T_{11}, T_{12}, \cdots, T_{1r}$ whose ϵ'_2 has at least one dash.

(b) Denote the set of terms in the basic set of MO elements as

$T_{21}, T_{22}, \cdots, T_{2s}$ whose ϵ''_2 has at least one dash.

(c) Detect if any term T_{1j}, $j \in \{1, 2, \ldots, r\}$ either subsumes or is subsumed by another term T_{2k}, $k \in \{1, 2, \ldots, s\}$.

If T_{1j} subsumes T_{2k}, then go to (d).

If T_{1j} is subsumed by T_{2k}, then go to (e).

Otherwise, stop.

(d) Record T_{1j} along with ϵ'_2 of T_{1j} and ϵ''_2 of T_{2k}. Then go to (f).

(e) Record T_{2k} along with ϵ'_2 of T_{1j} and ϵ''_2 of T_{2k}.

(f) Write down all terms which subsume either T_{1j} or T_{2k} along with their ϵ'_2 and ϵ''_2 parts.

Example: Apply the above algorithm to the basic set of MO elements in Table 3.3, to obtain MOPI A-T pairs, MOPI T-A pairs, MOCO A-T pairs, MOCO T-A pairs.

Set S_1: $(x_1 \quad x_2 \quad x_3)$ with $\epsilon'_2 = (- \; - \; 0)$.

Set S_2: $(x_3 \quad \bar{x}_4 \quad \bar{x}_5)$ with $\epsilon''_2 = (0 \; 0 \; -)$.

1. Applying procedure 1 to S_1 and S_2, one finds MOPI T-A pairs:

$$T = x_4 \, x_5 \qquad A = \bar{x}_4 \vee \bar{x}_5$$

ϵ_2^* for T is $(- \; 0 \; 0, \; 0 \; 0 \; -)$

ϵ_2^* for A is $(0 \; 0 \; 0, \; 0 \; 0 \; -)$

2. Applying procedure 2 to S_1 and S_2, one finds no MOPI T-A pair.

3. Applying procedure 3 to S_1 and S_2, one finds no MOCO A-T pair.

4. Applying procedure 5 to the basic set of MO elements one finds MOCO T-A pairs as follows:

	ϵ_2^*
$x_4 \, x_5$	$(- \; 0 \; 0, \; 0 \; 0 \; -)$
$x_1 \, \bar{x}_4$	$(- \; - \; 0, \; 0 \; 0 \; -)$
$x_2 \, \bar{x}_4$	$(- \; - \; 0, \; 0 \; 0 \; -)$
$\bar{x}_4 \, x_5$	$(0 \; - \; 0, \; 0 \; 0 \; -)$
$\bar{x}_4 \, \bar{x}_5$	$(- \; 0 \; 0, \; 0 \; 0 \; -)$
$\bar{x}_1 \, x_3 \, x_4$	$(- \; - \; 0, \; 0 \; 0 \; -)$
$\bar{x}_3 \, x_4 \, x_5$	$(- \; 0 \; 0, \; 0 \; 0 \; -)$
$x_1 \, \bar{x}_3 \, x_4 \, x_5$	$(- \; - \; -, \; 0 \; 0 \; -)$
$x_2 \, \bar{x}_3 \, x_4 \, x_5$	$(- \; - \; -, \; 0 \; 0 \; -)$
$\bar{x}_1 \, \bar{x}_2 \, x_4 \, x_5$	$(- \; 0 \; 0, \; 0 \; 0 \; -)$

The minimization procedure by the ILP method when cost-free inversions are available is about the same as the case when no inversion is available except the existence of some more inequalities which are generated from MOPI T-A pairs, MOPI A-T pairs, MOCO T-A pairs, and MOCO A-T pairs. Let

the set of those inequalities generated by all the MOPI T-A pairs, MOPI A-T pairs, MOCO T-A pairs and MOCO A-T pairs be denoted by $\mathbb{B}' \vec{y}'' \geq \vec{0}$.

Among all inequalities in $\mathbb{B}' \vec{y}'' \geq \vec{0}$, the generation of inequalities from MOPI T-A pairs is described in the following. The method of generation of other inequalities from MOPI A-T pairs, MOCO T-A pairs, and MOCO A-T pairs is similar and omitted here. Any function in the following procedure is a subset of the output functions $f_1, f_2, \cdots, f_m$.

Let T and A be the term and alterm of a MOPI T-A pair, respectively, where T implies $f_{j_1}, f_{j_2}, \cdots, f_{j_\ell}$ and A implies $f_{p_1}, f_{p_2}, \cdots, f_{p_q}$. Also, let the variable assignments to T be $y_{j_1}, y_{j_2}, \cdots, y_{j_\ell}$ and those to A be $y_{p_1}, y_{p_2}, \cdots, y_{p_q}$. Then the portion of the objective function corresponding to this MOPI T-A pair is

$$y_{j_1} + y_{j_2} + \cdots + y_{j_\ell} + y_{p_1} + y_{p_2} + \cdots + y_{p_q} + ct_3,$$

where c is the cost of the term T, and the inequalities corresponding to it are,

$$\begin{aligned}
y_{j_1} + y_{j_2} + \cdots + y_{j_\ell} - t_1 &\geq 0 \\
\ell t_1 - y_{j_1} - y_{j_2} - \cdots - y_{j_\ell} &\geq 0 \\
y_{p_1} + y_{p_2} + \cdots + y_{p_q} - t_2 &\geq 0 \\
q t_2 - y_{p_1} - y_{p_2} - \cdots - y_{p_q} &\geq 0 \\
t_1 + t_2 - t_3 &\geq 0 \\
2t_3 - t_1 - t_2 &\geq 0
\end{aligned}$$

where $y_{j_1}, y_{j_2}, \cdots, y_{j_\ell}, y_{p_1}, y_{p_2}, \cdots, y_{p_q}, t_1, t_2$, and t_3 are variables whose values are 1 or 0. t_1, t_2 and t_3 are the linearization variables of the term T, the alterm A, and the T-A pair, respectively.

The first inequality shows that if $y_{j_1} = y_{j_2} = \cdots = y_{j_\ell} = 0$, then $t_1 = 0$. The second inequality shows that if any one of $y_{j_1}, y_{j_2}, \cdots, y_{j_\ell}$ is 1, then $t_1 = 1$. Similarly are the third and the fourth inequalities.

The last two inequalities give $t_3 = 1$ if at least one of t_1 and t_2 is one, and $t_3 = 0$, otherwise. If $t_3 = 1$, then either T or A is used in a minimal network, or both. A cost $ct_3 = c$ is to be added to the network cost function. If $t_3 = 0$, neither T nor A is used. No cost of this MOPI T-A pair is to be added to the total cost.

Thus, six inequalities are generated for each of MOPI T-A pair, MOPI A-T pair, MOCO T-A pair, and MOCO A-T pair in the set of inequalities

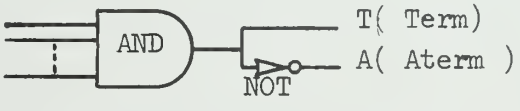
$$\mathbb{B}' \vec{y}'' \geq \vec{0}.$$

This set $\mathbb{B}' y'' \geq 0$ is appended to the ILP formulation $A_0 \vec{y}_0 \geq \vec{b}_0$ to form the ILP formulation when cost-free inversions are available.

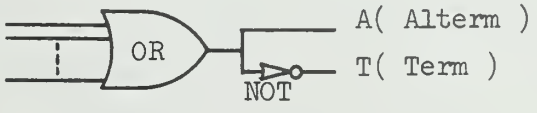
4.4 Minimization with NOT Gates Available

In the minimization when NOT gates are available, the procedure is the same as the one for the minimization with cost-free inversions in the last section, except the additional consideration of the cost of NOT gates.

The T-A pair and A-T pair mentioned in Sections 4.2 and 4.3 are realized by an AND gate and an OR gate with an additional NOT gate, respectively. This is depicted in the following pictures.



Realization of a T-A pair.



Realization of an A-T pair.

From the above pictures it is clear that in the realization of a T-A pair if A (an alterm) is never used for any output function in a network, the T-A pair is actually degenerated into a T (a term) and the inclusion of NOT gate (to realize the A) is superfluous. Similarly, the NOT gate in A-T pair realization is superfluous if the T (term) is never used in a network. The case that T (term) in T-A pair realization and A (alterm) in A-T pair realization are not used is assumed not to occur since if it does, a realization of lower cost (simply an OR gate and an AND gate, respectively) can always be used.

With the above consideration in mind, the formulation of the cost of a MOPI T-A pair, MOPI A-T pair, MOCO T-A pair and MOCO A-T pair is to be described. The generation of inequalities from MOPI T-A pairs, MOPI A-T pairs, MOCO T-A pairs and MOCO A-T pairs is the same as that given in the last section.

The portion of the objective function corresponding to a MOPI T-A pair (as illustrated in the last section) when NOT gate is used is modified to

$$y_{j_1} + y_{j_2} + \cdots + y_{j_\ell} + y_{p_1} + y_{p_2} + \cdots + ct_3 + 2\,(t_1 + t_2 - t_3)$$

where c is the cost of the structure of the term T of this MOPI T-A pair and 2 is the cost of a NOT gate (one gate-input and one gate) in the realization of the T-A pair.

The term $ct_3 + 2\,(t_1 + t_2 - t_3)$ assumes the value shown in the following.

t_1	t_2	t_3	$ct_3 + 2 \cdot (t_1 + t_2 - t_3)$	Remarks
0	0	0	0	T-A pair is not used.
0	1	1	c	No NOT gate is needed.
1	0	1	c	No NOT gate is needed.
1	1	1	c + 2	A NOT gate and an AND gate are counted.

The above table shows that cost c + 2 is added to the cost function of a network if and only if the T (term) and A (alterm) of the MOPI T-A pair are both used, i.e., $t_1 = t_2 = 1$. This is clear from the physical realization of a T-A pair.

The cost function for a MOPI A-T pair, MOCO A-T pair, MOCO T-A pair can be established similarly.

In conclusion, the inclusion of inverters (or inversions) increased the size of the ILP formulation because of the additional consideration of MOPI T-A pairs, MOPI A-T pairs, MOCO T-A pairs, and MOCO A-T pairs. In the case of minimization with cost-free inversions, the cost of any additional inverter is zero while it is 2 in the case of minimization with inverter of some cost. The inequalities generated from MOPI T-A pairs, MOPI A-T pairs, MOCO T-A pairs, and MOCO A-T pairs are the same in both cases.

5. MINIMIZATION WITH DON'T-CARES

5.1 Introduction

In the design of a switching network, given output functions are often incompletely specified, especially in design of multiple-output switching networks. There are certain input combinations for which the output can either be 1 or 0, or a certain number of input combinations never occur in some or all of the given output functions. The design of such multiple-output networks is known as the "don't-cares" problem. If a minimal multiple-output network is desired, the problem is a minimization problem with don't-cares.

A number of authors have investigated the minimization problem with don't-cares. McNaughton [24] defines a r-th order multiple-output prime implicant by taking the don't-care conditions into consideration. His method is primarily based on the concept of prime implicant and may be tedious and impractical if the number of output functions is large, because he practically considers all the products of all output functions and all the disjunctions of all output functions at the same time. McCluskey [22] treats don't-cares by assigning all 1 to find the prime implicants and then a minimal number of prime implicants are selected to construct a minimal multiple-output network. Su [36] uses don't-care conditions to compress a truth table to a compact form; then minimization is done, based on this compressed truth table. But no minimality is assured.

Several others have also treated the don't-care problems of a single output function. Chu [9] uses don't-cares to define the D-implicant and D-implication relation. Lawler [19] treats don't-cares by defining interval functions, and minimization is done with respect to the interval functions. Motts [27] uses don't-cares to define a weak prime implicant and then to solve the covering problem based on those weak prime implicants.

In this paper we will use don't-care conditions to find a minimal multiple-output network of AND-OR gates in at most two levels under the cost function defined in Chapter 1. Basically, in the minimization with don't-cares, the existence of MOPI terms, MOCO terms, MOPI alterms, and MOCO alterms is again taken into consideration. The only difference between the minimization problems with don't-cares and the minimization problems without don't-cares is in the procedure that generates the basic set of MO elements. Hence, this chapter will describe a systematic way to generate the smallest set of MO elements on which the minimization is based. The rest of the minimization procedures are the same as the case without don't-cares once the basic set of MO elements is found. It should be noted that we want to use the don't-care conditions on one hand to achieve the minimal realization, but on the other hand don't want to make the size of the basic set of MO elements too big since this will greatly increase the computational complexity in finding a minimal covering set for constructing a minimal multiple-output network.

The ILP method derived in Chapters 2 and 3 can be extended with modification to the synthesis of minimal multiple-output networks, no matter whether don't-cares are included or not. The design of a minimal multiple-output network when inversions are available (including the case of cost-free inversions and also the case of inversions of some cost) for incompletely specified output functions is a direct extension of that in Chapter 4.

5.2 Definitions

The set of given output functions $\{f_1, f_2, \ldots, f_m\}$ denoted by F is assumed to be incompletely specified in this chapter. Don't-cares occur in some or all of the functions in F. The occurrence of don't-cares in a function f_i may or may not be the same as that in another function f_j

for $j \neq i$, although some don't-cares may appear in every f_i for $i \in \{1, 2, \ldots, m\}$. The set of don't-cares of f_i will be denoted by d_i which also denotes a Boolean expression of the set. The collection of all d_i's is denoted by $D = \{d_1, d_2, \ldots, d_m\}$. Note that d_i in D is the counterpart of f_i in F. The union F U D is defined to mean the set $\{f_1 \vee d_1, f_2 \vee d_2, \ldots, f_m \vee d_m\}$.

In Section 2.2 the definitions were given for MOPI term, MOPI alterm, MOCO term, and MOCO alterm. In this section consideration will be given to two different types of MOPI and MOCO, in other words, the T-MOPI and the D-MOPI, and the T-MOCO and the D-MOCO, each of which is defined in the following.

Definition 5.2.1: A term is called a T-MOPI term of F if this term is a MOPI term of F and there exists no don't-care in D which implies this term. A term is a D-MOPI term if this term is a MOPI term of the union $F \cup D = \{f_1 \vee d_1, f_2 \vee d_2, \ldots, f_m \vee d_m\}$.

Definition 5.2.2: An alterm is called a T-MOPI alterm of F if this alterm is a MOPI alterm of F and is implied by no don't-care in D. An alterm is a D-MOPI alterm of F if this alterm is a MOPI alterm of the union of $F \cup D = \{f_1 \vee d_1, f_2 \vee d_2, \ldots, f_m \vee d_m\}$.

Definition 5.2.3: A term is called a T-MOCO term of F if this term is a MOCO term of F and is implied by no don't-care in D. A term is a D-MOCO term of F if this term is a MOCO term of the union F U D.

Definition 5.2.4: An alterm is called a T-MOCO alterm of F if this alterm is a MOCO alterm of F and is implied by no don't-care in D. An alterm is a D-MOCO alterm of F if this alterm is a MOCO alterm of the union F U D.

Definition 5.2.5: The collection of all T-MOPI terms of F is called T-basic set of MO elements of F. The collection of all D-MOPI terms of F is called D-basic set of MO elements of F. Similarly defined are the

T-sufficient set of MO elements and the D-sufficient set of MO elements of F.

Definition 5.2.6: Let Φ_i be either an OR-JOINT or an AND-JOINT Boolean form which defines an output function f_i, $f_i = \Phi_i$. Φ_i is called a T-Boolean form of f_i if $f_i = \Phi_i$ and there exists no don't-care condition in d_i which implies Φ_i. Φ_i is called a D-Boolean form of f_i if $f_i \vee d_i = \Phi_i$.

Definition 5.2.7: A network $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_m)$ is said to be a T-minimum cost network of outputs $f_1, f_2, \ldots, f_m$ if each Φ_i is a T-Boolean form of f_i for i = 1, 2, ..., m and there exists no other network $\eta\ (\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ which costs less than $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_m)$, where Φ'_i is also a T-Boolean form of f_i for i = 1, 2, ..., m.

Similarly, one can define a D-minimum cost network.

Definition 5.2.8: A network $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_m)$ is said to be a D-minimum cost network of outputs $f_1, f_2, \ldots, f_m$ if Φ_i is a D-Boolean form of f_i for i = 1, 2, ..., m and there exists no other network $\eta\ (\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ which costs less than $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_m)$ and every Φ'_i is also a D-Boolean form of f_i for i = 1, 2, ..., m.

The proper implication relation and the inclusion relation are also defined.

Definition 5.2.9: Let P and Q be two Boolean expressions (terms, alterms, or functions). Then P and Q are said to have proper implication relations if either P implies Q or Q implies P but $P \neq Q$.

Definition 5.2.10: A set S_2 is said to imply another set S_1 if each member of S_1 is implied by some member of S_2.

5.3 Theorems

In this section, some properties relating to the existence of T-MO elements and D-MO elements are investigated, where a T-MO element may be a T-MOPI term, a T-MOPI alterm, a T-MOCO term, or a T-MOCO alterm.

Similarly, a D-MO element may be a D-MOPI term, a D-MOPI alterm, a D-MOCO term, or a D-MOCC alterm.

If a function f_i is incompletely specified, its complemented function $\bar{f}_i$ is also incompletely specified. Both f_i and $\bar{f}_i$ have the same don't-care conditions denoted by d_i. Then it is clear that $f_i \cdot \bar{f}_i = d_i$ which is in contrast to the completely specified case where the right hand side of the equality is 0.

The above basic property of an incompletely specified function will be used later. In the following, a relationship between T-MO elements and D-MO elements is given first.

Theorem 5.3.1: For any T-MOPI term of F, there exists at least one D-MOPI term of F which is implied by the T-MOPI term.

Proof: Let T_i be this T-MOPI term with respect to the product of the subset of functions in F,

$$f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}.$$

By Definition 5.2.1,

$$T_i \text{ implies } f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}.$$

Let the don't-cares of the product of functions $f_{j_1} \cdot f_{j_2} \cdot \ldots \cdot f_{j_\ell}$ be denoted by d. Then, there exists at least one prime implicant of the disjunction $T_i \vee d$, say T'_i, such that

$$T'_i \text{ is implied by } T_i.$$

But according to the definition of the D-MOPI term, T'_i is a D-MOPI term. Hence, the statement of the theorem is true.

Similarly, the case that T_i is a T-MOPI term with respect to the product of functions

$$\bar{f}_{j_1} \cdot \bar{f}_{j_2} \cdot \ldots \cdot \bar{f}_{j_\ell}$$

can be proved.

Q.E.D.

In terms of T- and D-basic set of MO elements, we get the following theorem.

Theorem 5.3.2: The T-basic set of MO elements of F implies the D-basic set of MO elements of F.

Proof: According to Definition 5.2.5, the T-basic set of MO elements is the collection of all T-MOPI terms and the D-basic set of MO elements is the collection of all D-MOPI terms. But by Theorem 5.3.1 every T-MOPI term implies at least one D-MOPI term in the D-basic set of MO elements. Hence, the D-basic set of MO elements is implied by the T-basic set of MO elements.

Q.E.D.

Similar to Theorem 5.3.1, there are the following three theorems.

Theorem 5.3.3: For any T-MOPI alterm of F, there exists at least one D-MOPI alterm of F which is implied by the T-MOPI alterm.

Theorem 5.3.4: For any T-MOCO term of F, there exists at least one D-MOCO term of F which is implied by the T-MOCO term.

Theorem 5.3.5: For any T-MOCO alterm of F, there exists at least one D-MOCO alterm of F, which is implied by the T-MOCO alterm.

In summarizing the results of Theorems 5.3.1, 5.3.3 through 5.3.5, we get the following theorem.

Theorem 5.3.6: The T-sufficient set of MO elements of F implies the D-sufficient set of MO elements of F.

Similar to Theorem 2.4.2, there is the following theorem.

Theorem 5.3.7: Under the cost function defined in Section 1.3, there exists a D- (T-) minimum cost two-level network $\eta\ (\Phi_1, \Phi_2, \ldots, \Phi_m)$ realizing a given set of incompletely specified output functions $f_1, f_2, \ldots, f_m$, where $f_i \vee d_i = \Phi_i$ $(f_i = \Phi_i)$ for all $i = 1, 2, \ldots, m$ such that every Boolean form, either Φ_i or $\bar{\Phi}_i$, is expressed in disjunction of terms

and alterms contained in the D- (T-) sufficient set of MO elements of the set of output functions $F = \{f_1, f_2, \ldots, f_m\}$.

Some relations between a D-minimum cost network and a T-minimum cost network realizing the same set of output functions $F = \{f_1, f_2, \ldots, f_m\}$ are shown in the next two theorems.

Theorem 5.3.8: Let Φ_i be a D-Boolean form of f_i in a D-minimum cost network $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ and Φ'_i be a T-Boolean form of f_i in a T-minimum cost network $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$. Both $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$ and $\eta(\Phi'_1, \Phi'_2, \ldots, \Phi'_m)$ realize $F = \{f_1, f_2, \ldots, f_m\}$. Then each term or alterm in either Φ_i or $\bar{\Phi}_i$ is implied by some term or alterm in either Φ'_i or $\bar{\Phi}'_i$.

Proof: Assume that there is a term T_i in Φ_i which is not implied by any other term T_j, $j \neq i$, or alterm A_k in Φ'_i. Then, for some input combination, there is a case where

either the term $T_j = 1$ or the alterm $A_k = 1$, but

$T_i = 0$.

For this particular input combination one may have $\Phi'_i = 1$ and $\Phi_i = 0$. This contradicts the assumption that Φ_i is a D-Boolean form of the network $\eta(\Phi_1, \Phi_2, \ldots, \Phi_m)$, since from

$$\Phi_i = f_i \vee d_i \text{ and } \Phi'_i = f_i,$$

Φ'_i must imply Φ_i.

The other cases can be similarly proved.

Q.E.D.

From the above theorem, the following important result can be stated.

Theorem 5.3.9: If for a given set of incompletely specified functions $F = \{f_1, f_2, \ldots, f_m\}$ there exist a T-minimum cost network and a D-minimum cost network, both of which realize F, then the cost of the D-minimum cost network is not greater than the cost of the T-minimum cost network.

Theorem 5.3.9 gives the assurance that the solution of the minimization

problem with don't-cares will always yield a D-minimum cost network which has no greater cost than a T-minimum cost network realizing the same set of given output functions which are incompletely specified. This is all the reason why we want to study the minimization problem with don't-cares.

5.4 The Generation of D- and T-Basic Sets of MO Elements

This section describes the method of generating D- and T-basic sets of MO elements, which respectively are the collections of D- and T-MOPI terms for the given set of output functions $f_1, f_2, \ldots, f_m$ which are incompletely specified.

The generation of the T-basic set of MO elements for a given set of incompletely specified output functions is as follows:

(1) Set the functional values for those input combinations which are don't-cares to 0. This is done for every incompletely specified function. The set of output functions are then made completely specified.

(2) Apply the same generation procedure described in Chapter 3 to generate the basic set of MO elements for this set of output functions which are made completely specified.

The procedure is not repeated here. However, the method of forming the binary character ϵ^* corresponding to the above step (1) is given.

Let $\{f_1, f_2, \ldots, f_m\}$ be the set of output functions which are incompletely specified. One can form the binary character $\epsilon^* = (\epsilon_1 : \epsilon'_2, \epsilon''_2)$ for each input combination, where ϵ_1 is the identifier part and $\epsilon_2^* = (\epsilon'_2, \epsilon''_2)$ is the tag part. And,

ϵ_1: The sequence of 0 or 1 which represents the input combination. This is the same as that in Section 3.2.

ϵ'_2: Its i-th component is a dash (-) if ϵ_1 is a true input combination

or a don't-care of f_i, where i = 1, 2, ..., m.

ϵ''_2: Its i-th component is 0 if ϵ_1 is a true input combination or a don't-care of f_i. Its i-th component is a dash (-) if ϵ_1 is a false input combination of f_i, where i = 1, 2, ..., m.

After the formation of the binary character ϵ^* for every input combination, one can apply the McCluskey tabular method to obtain the T-basic set of MO elements.

The procedure for generating the D-basic set of MO elements consists of two steps. The first step is to find the baset set of MO elements for a set of completely specified functions which are obtained from the given set of incompletely specified output functions. This differs from that in the generation of T-basic set of MO elements in the above in the way of obtaining the completely specified functions from the given set of incompletely specified output functions. Here, the functional values of all don't-cares are set to 1 for both f_i (i.e., true input combinations) and $\bar{f}_i$ (i.e., false input combinations) while they are set to 0 for both f_i and $\bar{f}_i$ in the generation of the T-basic set of MO elements. The second step is to find a minimum set of MO elements to form the D-basic set of MO elements. This minimum set of MO elements must be implied by the T-basic set of MO elements.

For the first step, as before, we form the binary character ϵ^* for every input combination first, where $\epsilon^* = (\epsilon_1: \epsilon^*_2) = (\epsilon_1: \epsilon'_2, \epsilon''_2)$ and

ϵ_1: The same ϵ_1 part as that in the generation of T-basic set of MO elements.

ϵ'_2: Its i-th component is a dash (-) if the ϵ_1 part is either a true input combination or a don't-care of f_i, for i = 1, 2, ..., m. Otherwise, the i-th component is 0.

ϵ''_2: Its i-th component is a dash (-) if the ϵ_1 part is either a false

input combination or a don't-care of f_i for i = 1, 2, ..., m.

Otherwise, the i-th component is 0.

After the ϵ^* characters are formed, the McCluskey tabular method is applied to find the basic set of MO elements.

An example given in Table 5.1 at the end of this chapter illustrates the procedure stated above. The T-basic set of MO elements is shown in Table 5.2, and Table 5.4 shows the result of application of the first step above.

Next, one wants to find a minimum D-basic set of MO elements from the result obtained in the last step. This is done by a scheme similar to that used in the minimum covering problem. An implication table is first set up (as shown in Table 5.5) to find the implication relation between the members of the T-basic set of MO elements (represented by the unprimed alphabet letters on the horizontal top line) and the members of the MO elements in the result of application of the first step (represented by the primed alphabet letters on the left most column).

A cross (x) is placed in column k and row o' of the implication table if the following conditions hold:

(1) the term represented by k implies the term represented by o'.

(2) the ϵ_2^* part of the term o' has a dash (-) whenever the ϵ_2^* part of the term k has a dash in the corresponding position.

This is done for all columns and rows.

The implication table is to be simplified by the following procedures.

1. Row selection. A row is selected if this row is the only row that has a cross (x) in a column. Whenever a row is selected to be included in a solution, this row is deleted and all the columns which have crosses in that row are also deleted.

2. Column elimination. A column can be eliminated if there exists another

column such that for each row there is a cross in the column whenever there is a cross in the other column.

3. Row elimination. A row can be eliminated if there exists another row which has a cross in each column whenever there is a cross in this row of the same column.

The above procedure must be repeatedly applied until no more columns and rows can be eliminated or selected. If no more columns are left, the implication table is "simplified", otherwise, a cyclic table [22] results which can be solved by the following procedure.

The procedure for solving a cyclic table is described in the following, which is similar to but differs from that of McCluskey's in two points:

1. Starting from a column, an arbitrary row is selected in McCluskey's method, while we use the existence of the distinguished fundamental product to determine whether a row is to be selected or not.
2. Every row must be tried for selection, one at a time, in McCluskey's method while we terminate the procedure for a column once a row is selected.

Procedure for solving a cyclic table.

(a) Write down the cyclic table which is the remaining part of the implication table after row selection, column elimination and row elimination.

(b) Let us consider the first column of the cyclic table. One finds rows which have crosses in that column. For simplicity, assume that they are a', b', c', d', and so on, and without loss of generality, let d' imply c', c' imply b' which implies a'.

(c) Compare the ϵ_2^* parts of d' and c'. Detect positions in the $\epsilon_2^* = (\epsilon'_2, \epsilon''_2)$ parts of d' and c' which have dashes in the ϵ_2^* part of d' but not in the corresponding positions of c'. Let those positions

in ϵ'_2 portion of ϵ^*_2 part be i-th, j-th, etc., and that in the ϵ''_2 portion be k-th, ℓ-th, etc.

(d) Remove the fundamental products which are don't-cares of $f_i \cdot f_j \cdots$ and those of $\bar{f}_k \cdot \bar{f}_\ell \cdots$ from the set of fundamental products which imply the term d'. Check if any one of the remaining fundamental products which imply d' is a distinguished fundamental product. If yes, then d' is selected and go to step (e). (Steps (b), (c), (d) will be repeated for the next column.) Otherwise, d' is not considered. Then do steps (b), (c), (d) by treating d' and c' instead of c' and b', respectively. This is done for all a', b', c', ..., of this first column. Finally either some row is selected, or a' is selected if a' is the only one left after this step.

(e) Delete all columns which have crosses in the row just selected.

(f) Treat the next column by repeating steps (b), (c), (d) and (e), until no column is left, in which case the procedure terminates. The collection of all rows selected is the desired set of MO elements. In addition to this set, the set of MO elements obtained by "simplying" the implication table is then used to construct the D-basic set of MO elements. This is done by employing the following process.

a. List all MOPI terms which remained in the implication table, even after elimination or selection.

b. Check if any one of those MOPI terms is implied by some MOPI term of the MO elements set which are selected in the above procedure of simplying the implication table (including the cyclic table.) If yes, denote the former MOPI terms by o''s and the latter MOPI terms by q''s, otherwise, stop.

c. If the ϵ'_2 (or ϵ''_2) of q' has a dash whenever the corresponding position of ϵ'_2 (or ϵ''_2) part of o' has a dash, then select o'.

d. Combine these MOPI terms obtained in step c and the MO elements obtained from the implication table, the D-basic set of MO elements results.

For the example of Table 5.1, the implication table and the cyclic table are shown in Tables 5.5 and 5.6, respectively. Table 5.7 shows the solution of the implication table, and finally Table 5.8 gives the D-basic set of MO elements. The term with star sign (*) in Table 5.8 indicates that they are terms which are not included in Table 5.7.

After the D-basic set of MO elements is found, the minimization procedure follows in the same way as described in Chapters 3 and 4.

Table 5.1. Example

The following functions are given:

$f_1(x_1, x_2, x_3, x_4) = \Sigma(2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$

$f_2(x_1, x_2, x_3, x_4) = \Sigma(2, 3, 5, 6, 7, 10, 11, 14, 15)$

$f_3(x_1, x_2, x_3, x_4) = \Sigma(6, 7, 8, 9, 13, 14, 15)$

$d_1(x_1, x_2, x_3, x_4) = \Sigma(0, 1, 14)$

$d_2(x_1, x_2, x_3, x_4) = \Sigma(4, 8, 13)$

$d_3(x_1, x_2, x_3, x_4) = \Sigma(0, 1, 3, 11)$

	$\epsilon_1^{(i)}$				ϵ'_2			$\epsilon_2^{*(i)}$ ϵ''_2		
i	x_1	x_2	x_3	x_4	f_1	f_2	f_3,	$\overline{f}_1$	$\overline{f}_2$	$\overline{f}_3$
0	0	0	0	0	d	0	d	d	-	d
1	0	0	0	1	d	0	d	d	-	d
2	0	0	1	0	-	-	0	0	0	-
4	0	1	0	0	0	d	0	-	d	-
8	1	0	0	0	-	d	-	0	d	0
3	0	0	1	1	-	-	d	0	0	d
5	0	1	0	1	-	-	0	0	0	-
6	0	1	1	0	0	-	-	-	0	0
9	1	0	0	1	-	0	-	0	-	0
10	1	0	1	0	-	-	0	0	0	-
12	1	1	0	0	0	0	0	-	-	-
7	0	1	1	1	-	-	-	0	0	0
11	1	0	1	1	-	-	d	0	0	d
13	1	1	0	1	-	d	-	0	d	0
14	1	1	1	0	d	-	-	d	0	0
15	1	1	1	1	-	-	-	0	0	0

$\epsilon^* = (\epsilon_1 : \epsilon_2^*) = (\epsilon_1 : \epsilon'_2, \epsilon''_2)$

Table 5.1. (continued)

Table of ϵ^*- terms for Generating T-basic Set of MO Element

	$\epsilon_1^{(i)}$				ϵ'_2			ϵ''_2		
i	x_1	x_2	x_3	x_4	f_1	f_2	f_3	$\bar{f}_1$	$\bar{f}_2$	$\bar{f}_3$
0	0	0	0	0	0	0	0	0	-	0
1	0	0	0	1	0	0	0	0	-	0
2	0	0	1	0	-	-	0	0	0	-
4	0	1	0	0	0	0	0	-	0	-
8	1	0	0	0	-	0	-	0	0	0
3	0	0	1	1	-	-	0	0	0	0
5	0	1	0	1	-	-	0	0	0	-
6	0	1	1	0	0	-	-	-	0	0
9	1	0	0	1	-	0	-	0	-	0
10	1	0	1	0	-	-	0	0	0	-
12	1	1	0	0	0	0	0	-	-	-
7	0	1	1	1	-	-	-	0	0	0
11	1	0	1	1	-	-	0	0	0	0
13	1	1	0	1	-	0	-	0	0	0
14	1	1	1	0	0	-	-	0	0	0
15	1	1	1	1	-	-	-	0	0	0

Table 5.2.

T-Basic Set of MO Elements

Symbols	MOPI Terms	Fundamental Products	ϵ'_2 f_1	f_2	f_3	ϵ''_2 $\bar{f}_1$	$\bar{f}_2$	$\bar{f}_3$
a	x_3	(2, 3, 6, 7, 10, 11, 14, 15)	0	-	0	0	0	0
b	$x_1 x_4$	(9, 11, 13, 15)	-	0	0	0	0	0
c	$x_2 x_3$	(6, 7, 14, 15)	0	-	-	0	0	0
d	$x_2 x_4$	(5, 7, 13, 15)	-	0	0	0	0	0
e	$x_3 x_4$	(3, 7, 11, 15)	-	-	0	0	0	0
f	$x_1 \bar{x}_2$	(8, 9, 10, 11)	-	0	0	0	0	0
g	$\bar{x}_2 x_3$	(2, 3, 10, 11)	-	-	0	0	0	-
h	$x_1 x_2 x_4$	(13, 15)	-	0	-	0	0	0
i	$x_1 x_3 x_4$	(11, 15)	-	-	0	0	0	0
j	$x_2 x_3 x_4$	(7, 15)	-	-	-	0	0	0
k	$x_1 \bar{x}_2 x_3$	(10, 11)	-	-	0	0	0	-
l	$x_1 \bar{x}_3 x_4$	(9, 13)	-	0	-	0	-	0
m	$x_2 x_3 \bar{x}_4$	(6, 14)	0	-	-	-	0	0
n	$\bar{x}_1 x_2 x_4$	(5, 7)	-	-	0	0	0	0
o	$x_1 \bar{x}_2 \bar{x}_3$	(8, 9)	-	0	-	0	-	0
p	$x_2 \bar{x}_3 \bar{x}_4$	(4, 12)	0	0	0	-	-	-
q	$\bar{x}_1 x_2 x_4$	(4, 6)	0	0	0	-	0	0
r	$\bar{x}_1 x_2 \bar{x}_3$	(4, 5)	0	0	0	0	0	-
s	$\bar{x}_2 x_3 x_4$	(2, 10)	-	-	0	0	0	-
t	$\bar{x}_1 \bar{x}_2 x_3$	(2, 3)	-	-	0	0	0	0
u	$\bar{x}_2 \bar{x}_3 x_4$	(1, 9)	0	0	0	0	-	0
v	$\bar{x}_1 \bar{x}_2 \bar{x}_3$	(0, 1)	0	0	0	-	-	-
w	$\bar{x}_1 x_2 \bar{x}_3 x_4$	(5)	-	-	0	0	0	-
x	$\bar{x}_1 x_2 x_3 \bar{x}_4$	(6)	0	-	-	-	0	0
y	$x_1 \bar{x}_2 \bar{x}_3 x_4$	(9)	-	0	-	0	-	0
z	$x_1 x_2 \bar{x}_3 \bar{x}_4$	(12)	0	0	0	-	-	-

Table 5.3.

Table of ϵ^*-terms for Generating D-Basic Set of MO Elements

	$\epsilon_1^{(i)}$				ϵ'_2			ϵ''_2			
i	x_1	x_2	x_3	x_4	f_1	f_2	f_3	$\bar{f}_1$	$\bar{f}_2$	$\bar{f}_3$	Check marks**
0	0	0	0	0	-	0	-	-	-	-	✔
1	0	0	0	1	-	0	-	-	-	-	✔
2	0	0	1	0	-	-	0	0	0	-	✔
4	0	1	0	0	0	-	0	-	-	-	
8	1	0	0	0	-	-	-	0	-	0	
3	0	0	1	1	-	-	-	0	0	-	✔
5	0	1	0	1	-	-	0	0	0	-	
6	0	1	1	0	0	-	-	-	0	0	✔
9	1	0	0	1	-	0	-	0	-	0	✔
10	1	0	1	0	-	-	0	0	0	-	✔
12	1	1	0	0	0	0	0	-	-	-	
7	0	1	1	1	-	-	-	0	0	0	✔
11	1	0	1	1	-	-	-	0	0	-	✔
13	1	1	0	1	-	-	-	0	-	0	
14	1	1	1	0	-	-	-	-	0	0	✔
15	1	1	1	1	-	-	-	0	0	0	✔

** Check mark (✔) is placed whenever the row is combined with some other row.

Table 5.4.

Result of Step 1 of the Generation of D-Basic Set of MO Elements

Symbols	Terms	Fundamental Products	ϵ'_2	ϵ''_2
a'	x_3	(2, 3, 6, 7, 10, 11, 14, 15)	0 - 0	0 0 0
b'	x_4	(1, 3, 5, 7, 9, 11, 13, 15)	- 0 0	0 0 0
c'	$\bar{x}_2$	(0, 1, 2, 3, 8, 9, 10, 11)	- 0 0	0 0 0
d'	$x_1 x_3$	(10, 11, 14, 15)	- - 0	0 0 0
e'	$x_1 x_4$	(9, 11, 13, 15)	- 0 -	0 0 0
f'	$x_2 x_3$	(6, 7, 14, 15)	0 - -	0 0 0
g'	$x_2 x_4$	(5, 7, 13, 15)	- - 0	0 0 0
h'	$x_3 x_4$	(3, 7, 11, 15)	- - -	0 0 0
i'	$x_1 \bar{x}_3$	(8, 9, 12, 13)	0 0 0	0 - 0
j'	$x_2 \bar{x}_4$	(4, 6, 12, 14)	0 0 0	- 0 0
k'	$\bar{x}_1 x_2$	(4, 5, 6, 7)	0 - 0	0 0 0
l'	$\bar{x}_2 x_3$	(2, 3, 10, 11)	- - 0	0 0 -
m'	$\bar{x}_2 x_4$	(1, 3, 9, 11)	- 0 -	0 0 0
n'	$\bar{x}_3 \bar{x}_4$	(0, 4, 8, 12)	0 0 0	0 - 0
o'	$\bar{x}_2 \bar{x}_3$	(0, 1, 8, 9)	- 0 -	0 - 0
p'	$\bar{x}_1 \bar{x}_3$	(0, 1, 4, 5)	0 0 0	0 0 -
q'	$\bar{x}_1 \bar{x}_2$	(0, 1, 2, 3)	- 0 0	0 0 -
r'	$x_1 x_2 x_3$	(14, 15)	- - -	0 0 0
s'	$x_1 x_2 x_4$	(13, 15)	- - -	0 0 0
t'	$x_1 x_3 \bar{x}_4$	(10, 14)	- - 0	0 0 0
u'	$x_1 \bar{x}_3 x_4$	(9, 13)	- 0 -	0 - 0
v'	$x_2 x_3 \bar{x}_4$	(6, 14)	0 - -	- 0 0
w'	$\bar{x}_2 x_3 x_4$	(3, 11)	- - -	0 0 -
x'	$x_1 \bar{x}_2 \bar{x}_4$	(8, 10)	- - 0	0 0 0
y'	$\bar{x}_1 x_2 \bar{x}_4$	(4, 6)	0 - 0	- 0 0
z'	$x_2 \bar{x}_3 \bar{x}_4$	(4, 12)	0 0 0	- - -

Table 5.4. (continued)

a_1'	$\bar{x}_1\, x_2\, \bar{x}_3$	(4, 5)	0 - 0	0 0 -
b_1'	$\bar{x}_1\, \bar{x}_3\, x_4$	(1, 5)	- 0 0	0 0 -
c_1'	$\bar{x}_1\, \bar{x}_2\, x_4$	(1, 3)	- 0 -	0 0 -
d_1'	$\bar{x}_1\, \bar{x}_3\, \bar{x}_4$	(0, 4)	0 0 0	- - -
e_1'	$\bar{x}_1\, \bar{x}_2\, \bar{x}_3$	(0, 1)	- 0 -	- - -
f_1'	$x_1\, x_2\, \bar{x}_3\, x_4$	(13)	- - -	0 - 0
g_1'	$x_1\, x_2\, \bar{x}_3\, \bar{x}_4$	(12)	0 0 0	- - -
h_1'	$\bar{x}_1\, x_2\, \bar{x}_3\, x_4$	(5)	- - 0	0 0 -
i_1'	$x_1\, \bar{x}_2\, \bar{x}_3\, \bar{x}_4$	(8)	- - -	0 - 0
j_1'	$\bar{x}_1\, x_2\, \bar{x}_3\, \bar{x}_4$	(4)	0 - 0	- - -

Table 5.5. Implication Table

Members of the T-Basic Set of MO Elements in Table 5.2.

Members of the MO elements in Table 5.4.

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a'	x																									
b'		x	x																							
c'				x																						
d'					x																					
e'		x				x																				
f'			x																							
g'				x						x																
h'					x			x	x																	
i'																										
j'										x																
k'																										
l'							x		x							x	x									
m'																										
n'																										
o'															x			x		x						
p'																x										
q'																										
r'																										
s'							x																			
t'																										
u'											x											x				
v'												x											x			
w'																										
x'																										
y'																										
z'																										

Table 5.5. (continued)

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a_1'																		x								
b_1'																										
c_1'																										
d_1'																										
e_1'																						x				
f_1'																										
g_1'																										x
h_1'																							x			
i_1'																										
j_1'																										

Table 5.6. Cyclic Table

		b	h	q	r	term	ϵ'_2	ϵ''_2	FP
*	b'	x				x_4	- 0 0	0 0 0	(1, 3, 5, 7, 9, 11, 13, 15)
*	e'	x	x			$x_1 x_4$	- 0 -	0 0 0	(9, 11, 13, 15)
*	j'		x			$x_2 \bar{x}_4$	0 0 0	- 0 0	(4, 6, 12, 14)
*	p'			x		$\bar{x}_1 \bar{x}_3$	0 0 0	0 0 -	(0, 1, 4, 5)
	s'		x			$x_1 x_2 x_4$	- - -	0 0 0	(13, 15)
	y'			x		$\bar{x}_1 x_2 \bar{x}_4$	0 - 0	- 0 0	(4, 6)
	a_1'				x	$\bar{x}_1 x_2 \bar{x}_3$	0 - 0	0 0 -	(4, 5)

The following notations are used in solving the cyclic table, Table 5.6 which is described below.

1. FP: fundamental product.
2. DFP: distinguished fundamental product.
3. Circled numbers in FP column are those fundamental products which are don't-cares.

Steps in solving the above cyclic table, Table 5.6:

1. In column b, rows b' and e': The FP 11 is a don't-care for f_3 and FP 13 is a don't-care for f_2, so FP 11 and FP 13 are deleted. Neither FP 9 nor FP 15 is a DFP. Hence, b' is selected.
2. In column h, rows e' and s': The FP 13 in s' is a don't-care for f_2, so FP 13 is deleted. FP 15 in s' is not a DFP. Hence, e' is selected.
3. In column q, rows j' and y': The FP 4 in y' is a don't-care for f_2, so FP 4 is deleted. FP 6 in y' is not a DFP. Hence, j' is selected.
4. In column r, rows p' and a_1': The FP 4 in a_1' is a don't-care for f_2, so FP 4 is deleted. FP 5 in a_1' is not a DFP. Hence, p' is selected.

Table 5.7. The Solution of the Implication Table

Symbols	MOPI terms	Fundamental Products	ϵ'_2	ϵ''_2
a'	x_3	(2, 3, 6, 7, 10, 11, 14, 15)	0 - 0	0 0 0
c'	$\bar{x}_2$	(0, 1, 2, 3, 8, 9, 10, 11)	- 0 0	0 0 0
e'	$x_1 x_4$	(9, 11, 13, 15)	- 0 -	0 0 0
f'	$x_2 x_3$	(6, 7, 14, 15)	0 - -	0 0 0
g'	$x_2 x_4$	(5, 7, 13, 15)	- - 0	0 0 0
h'	$x_3 x_4$	(3, 7, 11, 15)	- - -	0 0 0
j'	$x_2 \bar{x}_4$	(4, 6, 12, 14)	0 0 0	- 0 0
l'	$\bar{x}_2 x_3$	(2, 3, 10, 11)	- - 0	0 0 -
o'	$\bar{x}_2 \bar{x}_3$	(0, 1, 8, 9)	- 0 -	0 - 0
p'	$\bar{x}_1 \bar{x}_3$	(0, 1, 4, 5)	0 0 0	0 0 -
u'	$x_1 \bar{x}_3 x_4$	(9, 13)	- 0 -	0 - 0
v'	$x_2 x_3 \bar{x}_4$	(6, 14)	0 - -	- 0 0
z'	$x_2 \bar{x}_3 \bar{x}_4$	(4, 12)	0 0 0	- - -
e_1'	$\bar{x}_1 \bar{x}_2 \bar{x}_3$	(0, 1)	- 0 0	- - -
h_1'	$\bar{x}_1 x_2 \bar{x}_3 x_4$	(5)	- - 0	0 0 -

Table 5.8. The D-Basic Set of MO Elements

MOPI terms	Fundamental Products	ϵ'_2	ϵ''_2	
x_3	(2, 3, 6, 7, 10, 11, 14, 15)	0 - 0	0 0 0	
x_4	(1, 3, 5, 7, 9, 11, 13, 15)	- 0 0	0 0 0	*
$\bar{x}_2$	(0, 1, 2, 3, 8, 9, 10, 11)	- 0 0	0 0 0	
$x_1 x_3$	(10, 11, 14, 15)	- - 0	0 0 0	*
$x_1 x_4$	(9, 11, 13, 15)	- 0 -	0 0 0	
$x_2 x_3$	(6, 7, 14, 15)	0 - -	0 0 0	
$x_2 x_4$	(5, 7, 13, 15)	- - 0	0 0 0	
$x_3 x_4$	(3, 7, 11, 15)	- - -	0 0 0	
$x_1 \bar{x}_3$	(8, 9, 12, 13)	0 0 0	0 - 0	*
$x_2 \bar{x}_4$	(4, 6, 12, 14)	0 0 0	- 0 0	
$\bar{x}_1 x_2$	(4, 5, 6, 7)	0 - 0	0 0 0	*
$\bar{x}_2 x_3$	(2, 3, 10, 11)	- - 0	0 0 -	
$\bar{x}_2 \bar{x}_3$	(0, 1, 8, 9)	- 0 -	0 - 0	
$\bar{x}_1 \bar{x}_3$	(0, 1, 4, 5)	0 0 0	0 0 -	
$\bar{x}_1 \bar{x}_2$	(0, 1, 2, 3)	- 0 0	0 0 -	*
$x_1 \bar{x}_3 x_4$	(9, 13)	- 0 -	0 - 0	
$x_2 x_3 \bar{x}_4$	(6, 14)	0 - -	- 0 0	
$x_1 \bar{x}_2 \bar{x}_4$	(8, 10)	- - 0	0 0 0	*
$\bar{x}_1 x_2 \bar{x}_4$	(4, 6)	0 - 0	- 0 0	*
$x_2 \bar{x}_3 \bar{x}_4$	(4, 12)	0 0 0	- - -	
$\bar{x}_1 x_2 \bar{x}_3$	(4, 5)	0 - 0	0 0 -	*
$\bar{x}_1 \bar{x}_3 x_4$	(1, 5)	- 0 0	0 0 -	*
$\bar{x}_1 \bar{x}_3 \bar{x}_4$	(0, 4)	0 0 0	- - -	*
$\bar{x}_1 \bar{x}_2 \bar{x}_3$	(0, 1)	- 0 -	- - -	
$\bar{x}_1 x_2 \bar{x}_3 x_4$	(5)	- - 0	0 0 -	

LIST OF REFERENCES

1. Abhyanker, S., "Minimal Sum of Products of Sum Expression of Boolean Functions", IRE Transaction on Electronic Computers, EC-7, pp. 268-275, December 1958.

2. Abhyanker, S., "Absolute Minimal Expressions of Boolean Functions", IRE Transaction on Electronic Computers, EC-8, pp. 3-8, March 1959.

3. Akers, S. B. Jr., "A Diagrammatic Approach to Multiple Level Logic Synthesis", IEEE Transaction on Electronic Computers, EC-14, pp. 174-181, April 1965.

4. Akers, S. B. Jr., "A Truth Table Method for the Synthesis of Combinational Logic", IRE Transaction on Electronic Computers, EC-10 pp. 604-615, December 1961.

5. Bartee, T. C., "Computer Design of Multiple Output Logical Networks", IRE Transaction on Electronic Computers, EC-10, pp. 21-30, March 1961.

6. Bartee, T. C., Lebow, I. L., Reed, I. S., Theory and Design of Digital Machine, McGraw Hill Book Company, N.Y., 1962.

7. Burke, R. E., and Van Bosse, J. G , "NAND-AND Circuits", IEEE Transaction on Electronic Computers, EC-14, pp. 63-65, February 1965.

8. Caldwell, S. H., Switching Circuits and Logical Design, John Wiley & Sons, Inc., N.Y., 1958.

9. Chu, J. T., "Some Methods for Simplifying Circuits Using Don't-Care Conditions", ACM Journal, pp. 497-512, 1961.

10. Cobham, A., Fridshal, R., and North, J. H., "An Application of Linear Programming to the Minimization of Boolean Functions", Symposium on Switching Theory and Logical Design, 1961.

11. Dietmeyer, D. L., and Schneider, P. R., "Identification of Symmetry, Redundance, and Equivalence of Boolean Functions", IEEE Transaction on Electronic Computers, EC-16, pp. 804-817, December 1967.

12. Ellis, D. E., "A Synthesis of Combinational Logic with NAND and NOR Elements", IEEE Transaction on Electronic Computers, EC-14, pp. 701-705, October 1965.

13. Ghazala, M. J., "Irredundant Disjunctive and Conjunctive Forms of a Boolean Function", IBM Journal, April 1957.

14. Gimpel, J. F., "The Minimization of TANT Networks", IEEE Transaction on Electronic Computers, EC-16, pp. 18-38, February 1967.

15. Gimpel, J. F., "A Redundant Technique for Prime Implicant Table", IEEE Transaction on Electronic Computers, EC-14, pp. 535-541, August 1965.

16. Harris, B., "An Algorithm for Determining Minimal Representation of Logical Function", IRE Transaction on Electronic Computers, EC-16, pp. 103-108, June 1957.

17. Hohn, F. E., Applied Boolean Algebra--An Elementary Introduction, The MacMillan Company, N.Y., 1966.

18. Karnaugh, M., "The Map Method for Synthesis of Combinational Logic Circuits", AIEE Transactions, Vol. 72, pp. 593-598, 1953.

19. Lawler, E. L., "An Approach to Multiple-Level Boolean Minimization", Journal of ACM, Vol. 11, No. 3, pp. 283-295, July 1964.

20. Maley, G. A., and Earle, J., "Synthesizing Multiple-Output Switching Networks", 7th Annual Symposium on Computer and Data Processings, Denver Research Institute, Estes Park, Colorado, July 28-29, 1960.

21. McCluskey, E. J., and Bartee, T. C., editors, Minimization Theory--A Survey of Switching Circuit Theory, McGraw Hill, N.Y., pp. 67-88, 1962.

22. McCluskey, E. J., Introduction to the Theory of Switching Circuits, McGraw Hill, N.Y., 1965.

23. McCluskey, E. J., and Schorr, H., "Essential Multiple-Output Prime Implicants", Symposium on Mathematical Theory of Automata, Polytechnic Institute of Brooklyn, April 1962.

24. McNaughton, R., and Mitchell, B., "The Minimality of Rectifier Net with Multiple Outputs Incompletely Specified", Journal of Franklin Institute, Vol. 264, No. 6, pp. 457-480, December 1957.

25. Meo, A. R., "On the Minimal 3rd Order Expression of a Boolean Function", AIEE Symposium on Switching Theory and Logical Design, Chicago, September 1962.

26. Miller, R. E., "Switching Theory", Vol. I., Combinational Circuits, John Wiley & Sons, Inc., N.Y.

27. Motts, T. H. Jr., "An Algorithm for Determining Minimal Normal Forms of an Incomplete Truth Function", IEEE Transaction on Electronic Computers, EC-10, pp. 73-76, March 1961.

28. Mueller, R. K., and Urbana, R. H., "A Topological Method for the Determination of the Minimal Forms of a Boolean Function", IRE Transaction on Electronic Computers, Vol. EC-5, No. 3., pp. 126-132, September 1959.

29. Phister, M. J. Jr., Logical Design of Digital Computer, John Wiley & Sons, Inc., N.Y., 1958.

30. Polansky, R. B., "Minimization of Multiple-Output Switching Circuits", AIEE Transaction on Communication and Electronics, pt. 1, Vol. 80, No. 53, pp. 67-73, March 1961.

31. Prather, R., Introduction to Switching Theory--A Mathematical Approach, Allyn and Bacon, Inc., Boston, Massachusetts, 1967.

32. Pyne, I. B., and McCluskey, E. J. Jr., "An Essay on Prime Implicant Tables", SIAM, No. 4, pp. 604-631, December 1961.

33. Quine, W. V., "A Way to Simplify Truth Functions", American Mathematics Monthly, Vol. 62, No. 9, pp. 627-631, November 1955.

34. Roth, J. P., "Minimization Over Boolean Trees", IBM Journal of Research and Development, No. 4, 5, pp. 543-558, November 1960.

35. Schneider, P. R., and Dietmeyer, D. L., "An Algorithm for Synthesis of Multiple-Output Combinational Logic", IEEE Transaction on Electronic Computers, EC-17, pp. 117-128, February 1968.

36. Su, Y. H., and Dietmeyer, D. L., "Computer Reduction of Two-Level Multiple-Output Switching Circuit", IEEE Transaction on Electronic Computers, EC-18, pp. 58-63, January 1969.

37. Torng, H. C., Introduction to the Logical Design of Switching Systems, Addison-Wesley Publishing Co., Inc., Reading Massachusetts.

38. Vandling, G. C., "The Simplification of Multiple-Output Switching Networks Composed of Unilateral Devices", IRE Transaction on Electronic Computers, EC-9, No. 4, pp. 477-486, December 1960.

39. Weiner, P., and Dwyer, T. F., "Discussion of Some Flaws in the Classical Theory of Two-Level Minimization of Multiple Output Switching Networks", IEEE Transaction on Electronic Computers, EC-17, No. 2, pp. 184-186, February 1968.

APPENDIX A

AN EXAMPLE FOR SECTION 2.3: HEURISTIC REMARKS

This is to illustrate why in the definition of a MOCO term it is not necessary either for a MOCO term to be a MOPI term of a product of some functions or for the complement of a MOCO term to be a MOPI alterm of a product of some complemented functions.

Let a minimum network be given for four output functions f_1, f_2, f_3, f_4

$$f_1 = \Phi_1 = \cdots \vee x_1 x_2 x_3 \vee x_1 \bar{x}_3 \vee \cdots \quad \text{OR-JOINT}$$

$$f_2 = \Phi_2 = \cdots \vee x_1 x_2 x_3 \vee x_1 \bar{x}_3 \vee \cdots \quad \text{OR-JOINT}$$

$$f_3 = \Phi_3 = \cdots (\cdots) (x_1 x_2 x_3) x_4 \quad \text{AND-JOINT}$$

$$f_4 = \Phi_4 = \cdots (\cdots) (x_1 x_2 x_3) x_4 \quad \text{AND-JOINT}$$

where the term $x_1 x_2 x_3$ is a MOCO term according to Definition 2.2.4.

From the above Boolean expressions of f_1 and f_2 it is clear that $x_1 x_2 x_3$ is not a MOPI term for f_1 and f_2 since the MOPI term of f_1 and f_2 is $x_1 x_2$ (or possibly, x_1 or x_2 because of other terms of f_1 or f_2). Similarly, from the Boolean expression of f_3 and f_4, $\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 = \overline{(x_1 x_2 x_3)}$ is not a MOPI alterm of $\bar{f}_3 \bar{f}_4$ since the MOPI alterm of $\bar{f}_3$ and $\bar{f}_4$ is $\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4$. Neither the use of the MOPI term $x_1 x_2$ nor the use of the complement of the MOPI alterm $\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4$ i.e., $x_1 x_2 x_3 x_4$, as a MOCO term will yield a network of lower cost for f_1, f_2, f_3, and f_4. This shows that a MOCO term does not have to be a MOPI term of a product of some functions and that the complement of a MOCO term does not have to be a MOPI alterm of a product of some complemented functions.

Similarly, one can show that a MOCO alterm does not have to be a MOPI alterm of some set of functions and the complement of a MOCO alterm does not have to be a MOPI term of some other set of the complement of functions.

APPENDIX B

BRANCH-AND-BOUND METHOD FOR FINDING AN OPTIMAL GROUPING PATTERN (SEE SECTION 3.4.4)

There are 2^m possible grouping patterns for a set of m output functions. It may be computationally infeasible to find an optimal grouping pattern if all of the 2^m possible grouping patterns have to be tested since each test requires many computations for solving the corresponding ILP problem. The scheme to be described here is first to find the smallest set of grouping patterns such that among them the optimal grouping pattern can be found. Then, a systematic testing of each grouping pattern in the smallest set by the branch-and-bound method which reduces the above computational difficulty to some extent is presented. The smallest set here simply means that no more grouping patterns can be removed from the set by any of the elimination rules to be described later.

The symbol G is used to denote the smallest set of grouping patterns which can be derived from the sufficient set of MO elements. The reason why the sufficient set of MO elements is used to generate the smallest set of grouping patterns G is as follows.

(1) Functions which are expressed in OR-JOINT Boolean form (AND-JOINT Boolean form) are correlated through the MOPI terms and MOPI alterms. For example, the functions

$f_1 = T_{11} \vee \cdots \vee \cdots$ (OR-JOINT)

$f_2 = T_{11} \vee \cdots \vee \cdots$ (OR-JOINT)

are correlated to each other through the term T_{11} which is a MOPI term.

(2) Functions expressed in OR-JOINT Boolean form are correlated to functions expressed in AND-JOINT Boolean form through the MOCO terms and MOCO alterms. For example, the functions

$f_1 = T_{11} \vee A_{12} \vee \ldots \vee \ldots$ (OR-JOINT)

$f_3 = (A_{12})\ (T_{11})\ (\ldots)\ \ldots$ (AND-JOINT)

are correlated to each other through the term T_{11} and alterm A_{12} which are MOCO term and MOCO alterm, respectively.

The optimal grouping pattern is likely to be the one such that the functions having the Boolean forms represented by this optimal grouping pattern are correlated through the largest number of MOPI terms, MOPI alterms, MOCO terms and MOCO alterms. Since the sufficient set of MO elements contains all the MOPI terms, MOPI alterms and MOCO terms and MOCO alterms excluding those MOPI alterms which satisfy the condition of Theorem 2.4.1, so the smallest set of grouping patterns G can be generated from the sufficient set of MO elements.

In the following, two grouping patterns will be generated from each MOPI term and MOPI alterm, and one from each MOCO term and MOCO alterm. Rules for reduction of the number of grouping patterns generated from MOPI terms, MOPI alterms, MOCO terms, or MOCO alterms will be given.

The notation PI_{ij} is used to denote a MO element which is either a MOPI term or a MOPI alterm in Definition B.1, and is either a MOCO term or a MOCO alterm in Definition B.2. The tag part of the binary character of PI_{ij} is denoted by $\epsilon_2^*(i, j) = (\epsilon'_2(i, j), \epsilon''_2(i, j))$. However, PI_{ij} and $\epsilon_2^*(i, j)$ will be denoted simply as PI and ϵ_2^*, respectively, if they are independent of i and j.

Definition B.1: Let PI_{ij} be either a MOPI term or a MOPI alterm, then the grouping patterns generated by PI_{ij} are $\vec{\sigma}'(i, j)$ and $\vec{\sigma}''(i, j)$.

$\vec{\sigma}'(i, j) = (\sigma'_1(i, j), \sigma'_2(i, j), \ldots, \sigma'_m(i, j))$

$\vec{\sigma}''(i, j) = (\sigma''_1(i, j), \sigma''_2(i, j), \ldots, \sigma''_m(i, j))$,

where

σ'_k (i, j) = 1, if the k-th component of ϵ'_2 (i, j) of PI_{ij} is a dash (-).

= -, otherwise.

σ''_k (i, j) = 0, if the k-th component of ϵ''_2 (i, j) of PI_{ij} is a dash (-).

= -, otherwise.

The $\vec{\sigma}'$ (i, j) corresponds to a grouping pattern in which the output functions whose corresponding components in $\vec{\sigma}'$ (i, j) are 1 are correlated through PI_{ij} and are preset to OR-JOINT Boolean forms. The $\vec{\sigma}''$ (i, j) corresponding to a grouping pattern in which the output functions whose corresponding components in $\vec{\sigma}''$ (i, j) are 0 are correlated through PI_{ij} and are preset to AND-JOINT Boolean forms.

For example, if a PI_{ij} has ϵ_2 (i, j) = (- - 0 0, 0 0 0 -), then two grouping patterns generated by PI_{ij} are $\vec{\sigma}'$ (i, j) = (1 1 - -) and $\vec{\sigma}''$ (i, j) = (- - -0). The grouping pattern $\vec{\sigma}'$ = (1 1 - -) for PI_{ij} means that this PI_{ij} may be shared by f_1 and f_2 which are in OR-JOINT Boolean form, and the grouping pattern $\vec{\sigma}''$ (i, j) = (- - - 0) for PI_{ij} means that this PI_{ij} may be used by f_4 only which is in AND-JOINT Boolean form.

Since corresponding to each MOPI term and MOPI alterm in the sufficient set of MO elements two grouping patterns are generated, the number of such grouping patterns is naturally large. However, many of them can be eliminated according to the following rules.

Elimination Rules:

Rule 1. Let $\vec{\sigma}'$ (i, j) (or $\vec{\sigma}''$ (i, j)) be a grouping pattern generated from a MOPI term or a MOPI alterm of the sufficient set of MO elements. Then $\vec{\sigma}'$ (i, j) can be eliminated if there exists another grouping pattern $\vec{\sigma}'$ (k, ℓ) (or $\vec{\sigma}''$ (k, ℓ)) generated from another MOPI term

or MOPI alterm of the same sufficient set of MO elements such that:

1. Whenever a component of $\vec{\sigma}'(i, j)$ (or $\vec{\sigma}''(i, j)$) is 1 (or 0), the corresponding component of $\vec{\sigma}'(k, \ell)$ (or $\vec{\sigma}''(k, \ell)$) is also 1 (or 0), and
2. Whenever there is a dash (-) in a component of $\vec{\sigma}'(i, j)$ (or $\vec{\sigma}''(i, j)$), the corresponding component of $\vec{\sigma}'(k, \ell)$ (or $\vec{\sigma}''(k, \ell)$) is a dash (-) or 1 (or 0). This must be true for every component.

For example, if $\vec{\sigma}'(i, j) = (1\ 1\ -\ -)$ and $\vec{\sigma}'(k, \ell) = (1\ 1\ -\ 1)$, then $\vec{\sigma}'(i, j)$ can be eliminated.

It should be noted that in the above Rule 1, the grouping pattern $\vec{\sigma}'(k, \ell)$ (or $\vec{\sigma}''(k, \ell)$) has not fewer 1 (or 0) than that of $\vec{\sigma}'(i, j)$ (or $\vec{\sigma}''(i, j)$). Accordingly, there are no fewer output functions in the grouping pattern $\vec{\sigma}'(k, \ell)$ (or $\vec{\sigma}''(k, \ell)$) than that in $\vec{\sigma}'(i, j)$ (or $\vec{\sigma}''(i, j)$) which share a MO element. Thus, the grouping pattern $\vec{\sigma}'(i, j)$ (or $\vec{\sigma}''(i, j)$) can be eliminated.

In Table B.1, a set of grouping patterns are generated from the sufficient set of MO elements obtained in the example of Section 3.2. The grouping patterns $\vec{\sigma}'(i, j)$ generated from the MOPI terms in Table B.1 are all eliminated except (1 1 1) which is generated from $x_1 \bar{x}_3 x_4 x_5$ The grouping patterns $\vec{\sigma}''(i, j)$ are all eliminated except (- 0 0) which is generated from $\bar{x}_1 \bar{x}_2 \bar{x}_3 \bar{x}_5$.

Definition B.2: The grouping patterns generated by a MOCO term or a MOCO alterm PI_{ij} are denoted by

$$\vec{\sigma}'''(i, j) = (\sigma'''_1(i, j), \sigma'''_2(i, j), \ldots, \sigma'''_m(i, j)),$$

where

σ'''_k (i, j) = 1 if the k-th component of ϵ'_2 (i, j) of PI_{ij} is a dash (-).

= 0 if the k-th component of ϵ''_2 (i, j) of PI_{ij} is a dash (-).

= - if both the k-th components of ϵ'_2 (i, j) and ϵ''_2 (i, j) of PI_{ij} are 0's.

For example, the grouping pattern σ''' (i, j) = (1 1 0) in Table B.1 is generated from the MOCO alterm $x_1 \vee x_2 \vee x_3$ whose ϵ'_2 (i, j) = (- - 0) and ϵ''_2 (i, j) = (0 0 -).

It should be noted that the $\vec{\sigma}'''$ (i, j) in Definition B.2 corresponds to a grouping pattern in which the output functions whose corresponding component in $\vec{\sigma}'''$ (i, j) is either 1 or 0 share the PI_{ij} from which the $\vec{\sigma}'''$ (i, j) is generated. The output functions whose corresponding component in $\vec{Q}'''$ (i, j) is 1 are preset to OR-JOINT Boolean form and the output functions whose corresponding component in $\vec{\sigma}'''$ (i, j) is 0 are preset to AND-JOINT Boolean form.

The Elimination Rule 1 can also apply to the set of grouping patterns $\vec{\sigma}'''$ (i, j) which are generated from either MOCO terms or MOCO alterms. It is restated as Elimination Rule 2.

Rule 2. Let $\vec{\sigma}'''$ (i, j) and $\vec{\sigma}'''$ (k, ℓ) be two grouping patterns generated from either MOCO terms or MOCO alterms. Then $\vec{\sigma}'''$ (i, j) can be eliminated if the component of $\vec{\sigma}'''$ (k, ℓ) is 1 (or 0) whenever the corresponding component of $\vec{\sigma}'''$ (i, j) is 1 (or 0), and whenever there is a component of $\vec{\sigma}'''$ (i, j) is a dash (-), the corresponding component of $\vec{\sigma}'''$ (k, ℓ) is 1 (or 0) or a dash (-). This must be true for every component of $\vec{\sigma}'''$ (i, j) and $\vec{\sigma}'''$ (k, ℓ).

The reason why the grouping pattern $\vec{\sigma}'''$ (i, j) in the above Rule 2 can be eliminated is as follows. There are no fewer output functions which have 1 or 0 in the grouping pattern $\vec{\sigma}'''$ (k, ℓ) than that in the grouping pattern $\vec{\sigma}'''$ (i, j) to share a PI_{ij} (either a MOCO term or a MOCO alterm.) Hence, the consideration of $\vec{\sigma}'''$ (i, j) is not necessary and $\vec{\sigma}'''$ (i, j) can be eliminated.

Table B.1

Grouping Patterns Generated from the Sufficient Set of MO elements of the Example in Section 3.2.

PI_{ij}	$\vec{\sigma}'$ (i, j)	$\vec{\sigma}''$ (i, j)	$\vec{\sigma}'''$ (i, j)	Type of MO element
x_1	(1 1 -)	(- - -)		MOPI term
x_2	(1 1 -)	(- - -)		MOPI term
x_3	(1 1 -)	(- - 0)		MOPI term
$\bar{x}_4$	(- - -)	(- - 0)		MOPI term
$\bar{x}_5$	(- - -)	(- - 0)		MOPI term
$\bar{x}_1\ \bar{x}_2$	(- - -)	(- - 0)		MOPI term
$\bar{x}_4\ \bar{x}_5$	(- 1 -)	(- - 0)		MOPI term
$x_4\ x_5$	(1 - -)	(- - -)		MOPI term
$\bar{x}_4\ \bar{x}_5$	(1 - -)	(- - 0)		MOPI term
$x_1\ \bar{x}_4$	(1 1 -)	(- - 0)		MOPI term
$x_2\ \bar{x}_4$	(1 1 -)	(- - 0)		MOPI term
$\bar{x}_1\ \bar{x}_2\ \bar{x}_3\ x_4$	(- - -)	(- 0 0)		MOPI term
$\bar{x}_1\ \bar{x}_2\ \bar{x}_3\ \bar{x}_5$	(- - -)	(- 0 0)*		MOPI term
$\bar{x}_1\ \bar{x}_2\ x_4\ x_5$	(1 - -)	(- - 0)		MOPI term
$x_2\ \bar{x}_3\ x_4\ x_5$	(1 1 1)	(- - -)		MOPI term
$x_1\ \bar{x}_3\ x_4\ x_5$	(1 1 1)*	(- - -)		MOPI term
$x_1 \vee x_2 \vee x_3$			(1 1 0)*	MOCO alterm
$x_4\ x_5$			(1 - 0)	MOCO term

*Grouping patterns which are not eliminated by Elimination Rules 1 and 2.

For example, the grouping patterns generated from the MOCO alterm $x_1 \vee x_2 \vee x_3$ and the MOCO term $x_4\ x_5$ in Table B.1 are (1 1 0) and (1 - 0), respectively. (1 - 0) is eliminated according to the above Rule 2.

In Table B.1 there are only three grouping patterns which are not eliminated and marked with stars (*) after the Elimination Rules 1 and 2 are applied to the grouping patterns $\vec{\sigma}'(i, j)$, $\vec{\sigma}''(i, j)$ and $\vec{\sigma}'''(i, j)$.

For simplicity sake, the notation $\vec{\sigma}(i, j)$ is used to represent either $\vec{\sigma}'(i, j)$, $\vec{\sigma}''(i, j)$, or $\vec{\sigma}'''(i, j)$. $\vec{\sigma}(i, j)$ will be used in the rest of this appendix. It is understood that $\vec{\sigma}(i, j)$ is generated from either a MOPI term, MOPI alterm, MOCO term, or a MOCO alterm.

More grouping patterns can be eliminated by using the compatibility defined in the following.

Definition B.3: Let $\theta = \{\vec{\sigma}(i_1, j_1), \vec{\sigma}(i_2, j_2), \ldots, \vec{\sigma}(i_k, j_k)\}$ be a set of grouping patterns. θ is called a compatible set of grouping patterns if and only if there exists no component which has 1 (or 0) in a grouping pattern and 0 (or 1) in another grouping pattern of θ.

For example, $\theta = \{(1 - - 0\ 0), (- 0\ 1\ 0\ 0)\}$ is a compatible set of grouping patterns.

For each compatible set of grouping patterns, a new grouping pattern can be generated which is a representative of the compatible set. $\vec{\sigma}(p, q)$, a representative of a compatible set of grouping patterns $\theta = \{\vec{\sigma}(i_1, j_1), \vec{\sigma}(i_2, j_2), \ldots, \vec{\sigma}(i_k, j_k)\}$ is defined by:

(1) the r-th component of $\vec{\sigma}(p, q)$ is 1 (or 0) if either all of the r-th components of the grouping patterns in θ are 1 (or 0), or the r-th components of the grouping patterns in θ are either dash or 1 (or 0).

(2) the r-th component of $\vec{\sigma}(p, q)$ is a dash if all of the r-th components of the grouping patterns in θ are dashes.

(3) each r-th component is defined for r = 1, 2, ..., m by the above (1) and (2).

For example, let θ = {(1 - - 0 0), (- 0 1 0 0)} be a compatible set of grouping patterns, then the representative grouping pattern of θ is (1 0 1 0 0).

From the formation of the representative grouping pattern of a compatible set of grouping patterns θ and from the above example, we see that each grouping pattern of θ is only a part of its representative grouping pattern if the dash components are ignored. Thus, once the representative grouping pattern $\vec{\sigma}$ (p, q) of a compatible set of grouping patterns is formed, the whole compatible set can be eliminated. This is stated in the Elimination Rule 3.

Rule 3. Form the compatible sets of grouping patterns from the set of $\vec{\sigma}$ (i, j)'s. Find the representative of each compatible set. All grouping patterns in a compatible set are replaced by the representative of that compatible set of grouping patterns. This is done for every compatible set.

Let the collection of all grouping patterns remaining so far be denoted by G. This is the smallest set of grouping patterns mentioned before. The formation of the set G is summarized in the following.

1. Generate all grouping patterns from the MOPI terms and the MOPI alterms of the sufficient set of MO elements.
2. Apply the Elimination Rule 1 to eliminate some of the grouping patterns $\vec{\sigma}'$ (i, j) and $\vec{\sigma}''$ (i, j) stated in the rule.
3. Generate all grouping patterns from the MOCO terms and MOCO alterms of the sufficient set of MO elements.
4. Apply the Elimination Rule 2 to eliminate some of the grouping patterns $\vec{\sigma}'''$ (i, j) stated in the rule.
5. Form the compatible set of grouping patterns among those grouping patterns remaining after step 2 and step 4.

6. Generate the representative grouping pattern of each compatible set.
7. Apply the Elimination Rule 3. Replace the compatible set of grouping patterns by their representatives.
8. Collect all the grouping patterns remaining so far. This is the set G.

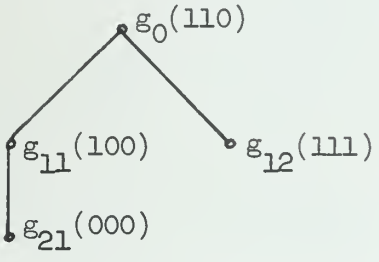
For example, the set G of Table B.1 is {(1 1 0), (- 0 0), (1 1 1)}.

After G set is formed, the optimal grouping pattern is found by the following process.

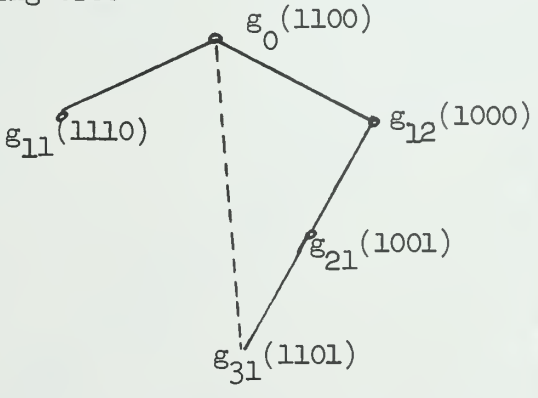
1. Expand each grouping pattern in G which has dashes in some k components into 2^k grouping patterns by placing 1 or 0 in those k components, where $1 \leq k \leq m$. Let this new collection be denoted by G'.
2. Arrange all grouping patterns of G' in a tree shape as follows.
   - 2.1. Every node of the tree represents one (and only one) grouping pattern, and no grouping pattern is represented by two or more nodes in the tree.
   - 2.2. A node is represented by symbol g_{ij} except the one which is denoted by g_0. g_0 is called the root of the tree which is placed at the top.
   - 2.3. The nodes g_{ij} of different j are placed in the same level. i denotes the i-th level from the root. j denotes the order from the left of that level.
   - 2.4. Two nodes g_{ij} and $g_{(i+1)k}$ for some j and k are connected by a line (i.e., the branch of the tree) if the grouping patterns represented by g_{ij} and $g_{(i+1)k}$ differ in only one (or, the smallest number of) component(s).
   - 2.5. If there are more than one node, $g_{i_1 j_1}$, $g_{i_2 j_2}$, ..., $g_{i_k j_k}$ for $k \geq 2$ connecting to the same node g_{rs} where $r > i_t$, $t \in \{1, 2, \ldots, k\}$, then disconnect all others except one

$g_{i_\ell j_\ell}$ where $(r\text{-}i_\ell)$ is the minimum over $\ell \in \{1, 2, \ldots, k\}$. If there are more than one such i_ℓ, then choose an arbitrary one, and disconnect the rest.

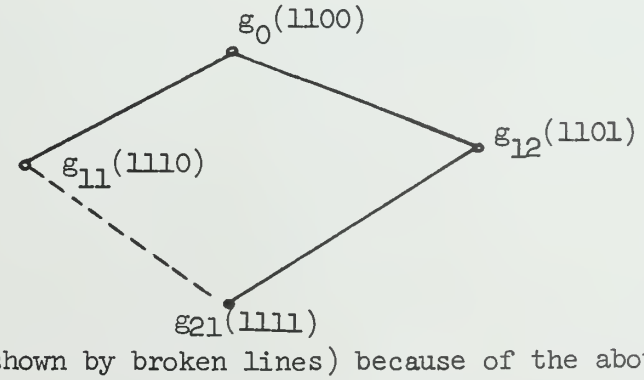
In the last example, G = {(1 1 0), (- 0 0), (1 1 1)}. (- 0 0) is expanded into (1 0 0) and (0 0 0). Thus, G' = {(1 1 0), (1 0 0), (0 0 0), (1 1 1)} which is arranged in the following tree.



The above process 2.5 is necessary in order to avoid the repetition of testing of each grouping pattern. For example, the branch connecting g_0 and g_{31} of the following tree



and the branch connecting g_{11} and g_{21} of the following tree



are not connected (shown by broken lines) because of the above process 2.5.

3. Let us term the ILP problem formulated according to the method described

in Section 3.4.2 and based on the specified grouping pattern represented by the node g_{ij} as "the ILP problem w.r.t. (with respect to) g_{ij}." Let us formulate an ILP problem w.r.t. g_o first and solve it. Denote the value of the objective function of its optimal solution as $c\ (g_0)$. Define the lowest bound of a network cost c_b as the lowest network cost found so far. Set $c_b = c\ (g_0)$.

4. Set i = 1 and j = 1.

5. A grouping pattern represented by g_{ij} is then tested by the following means. Since g_0 and g_{1j} has only one component which is different, with no loss of generality, let it be 0 in the k-th component of g_0 and 1 in that of g_{1j}.

Let the covering coefficient matrix $\mathbb{A}_0$ of the ILP problem w.r.t. g_0 be

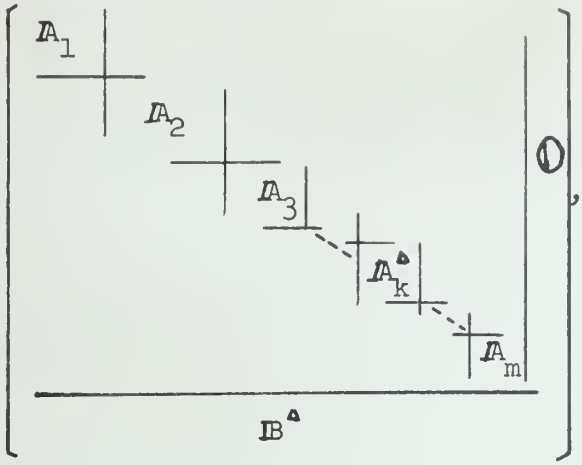
(5.1)
$$\begin{bmatrix} \mathbb{A}_1 & & & & & \\ & \mathbb{A}_2 & & & & \\ & & \mathbb{A}_3 & & & \\ & & & \mathbb{A}_k & & \\ & & & & \mathbb{A}_m & \\ \mathbb{B} & & & & & \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \mathbb{0} \end{matrix},$$

and the optimal solution vector be

(5.2)
$$\begin{bmatrix} \vec{y}_1 \\ \vdots \\ \vec{y}_k \\ \vdots \\ \vec{y}_m \\ \vec{t} \end{bmatrix}.$$

Then the covering coefficient matrix $\mathbb{A}_0$ of the ILP problem w.r.t. g_{1j} is

(5.3)



where only $\mathbb{A}_k^{\Delta}$ and $\mathbb{B}^{\Delta}$ are different from those in the matrix (5.1). $\mathbb{A}_k^{\Delta}$ is the covering matrix for f_k with the k-th component of this g_{1j} equal to 1. $\mathbb{B}^{\Delta}$ is the modified $\mathbb{B}$ due to the change of $\mathbb{A}_k$ to $\mathbb{A}_k^{\Delta}$.

Let the ILP problem with $\mathbb{A}_k \vec{y}_k \geq 1$ have a solution $\vec{y}'_k$. This does not necessarily be an optimal solution. Then the initial solution of the ILP problem w.r.t. g_{1j} is set to

$$\begin{bmatrix} \vec{0} \\ \vec{y}_1 \\ \vdots \\ \vec{y}'_k \\ \vdots \\ \vec{0} \\ \vec{y}_m \\ \vec{0} \\ t \end{bmatrix}$$

where only the $\vec{y}_k^0$ vector in (5.2) is replaced by $\vec{y}'_k$. From this solution (5.4) an optimal solution of the ILP problem w.r.t. g_{1j} is obtained and written as:

$$\begin{bmatrix} \vec{y}''_1 \\ \vdots \\ \vec{y}''_k \\ \vdots \\ \vec{y}''_m \\ t'' \end{bmatrix}$$

whose objective function has a value $c\ (g_{1j})$. Then set

$$c_b = c\ (g_{1j}) \text{ if } c\ (g_{1j}) < c_b$$

$$c_b \text{ unchanged, otherwise.}$$

6. Set j to j + 1, and repeat step 5 until all nodes in the i-th level of the tree are tested.

7. Set i = 2. Repeat steps 5 and 6 by treating g_{1j} as g_0 and g_{2k} as g_{1j} where g_{1j} and g_{2k} are connected by a branch in the tree. This is done until all nodes in this level are tested.

8. Repeat steps 5, 6, and 7 until all levels are tested. Let the final lowest cost bound $c_b = c\ (g_{pq})$ where g_{pq} is one of the nodes in the tree. Then the optimal grouping pattern is the grouping pattern represented by the node g_{pq}. The minimal network is the optimal solution of the ILP problem w.r.t. g_{pq}.

VITA

Frank Tuan-Lin Chen was born in Fuchow, Fuchien, China in June 1941. He graduated from the Taiwan Provincial Taipei Institute of Technology, Taipei, Taiwan, in Electrical Engineering in June 1961. After his graduation he served in Chinese Air Forces as a reserved officer from July 1961 to September 1962. From September 1962 to May 1964 he worked in the AIR ASIA Airline, Inc. In September 1964 he began his graduate study at the Iowa State University, Ames, Iowa where he received his MSEE degree in November 1965.

Since June 1966, he has been working as a research assistant with the project of ILLIAC IV computer in the Department of Computer Science, University of Illinois, Urbana.

He has been a member of Sigma Xi, the Association for Computing Machinery, and Institute of Electrical and Electronics Engineers.